The indispensable guide to all aspects of Web design — now in **FULL COLOR!**

Web Design

FOR

DUMMIES®

2nd Edition

Lisa Lopuck

Web Design

FOR

DUMMIES®

2ND EDITION

by Lisa Lopuck

Wiley Publishing, Inc.

Web Design

FOR

DUMMIES®

2ND EDITION

Web Design

FOR

DUMMIES®

2ND EDITION

by Lisa Lopuck

Wiley Publishing, Inc.

About the Author

In 1988, **Lisa Lopuck** got her first glimpse of multimedia while still at UCLA pursuing her degree in design. She saw a tiny, black-and-white, interactive HyperCard stack designed by The Voyager Company and immediately knew her career path.

Her first job out of school was working at the Apple Multimedia Lab in San Francisco. She then moved on to Skywalker Ranch, working with George Lucas to design educational CD-ROMs. The rest has been inter-active history — working with everyone from Kaleida to eBay, writing bestselling books, teaching, and speaking at conferences along the way.

In 1996, she co-founded Electravision, an award-winning Web design agency in San Francisco with clients such as Twentieth Century Fox, National Geographic, Microsoft, and Mall of America. Electravision's work won awards, including Yahoo!'s Best of the Year, Site of the Day, and Best of the Web. Electravision also created the highly acclaimed online murder mystery series, Suspect, one of the Web's first online entertainment series.

Recently, Lisa was an Associate Creative Director and Senior Producer at Disney where she managed the production of large-scale, large-budget, multi-language Web sites for Disney's theme parks and hotels around the world such as `www.DisneyCruiseLine.com` and `www.HongKongDisneyland.com`. Lisa is now a design consultant helping companies shape their Web creative strategy. She is also an accomplished watercolor artist, and a signature member of the National Watercolor Society. You can see her paintings online at `www.lopuck.com`, and her Web portfolio at `www.lopuck.com/portfolio`.

Dedication

For my husband, Matt, who is my chief evangelist, and for my daughter, Jasmine, who is now inspired to write and illustrate her own books.

For my parents who always encouraged me to reach for the stars.

Author's Acknowledgments

Thanks to the many people who helped make this book possible: The team at WebAssist who provided support and examples in this book, and awesome software that allows us non-techies to do great things; to colleagues and companies who provided wonderful examples to use in this book; to my friends at Macromedia and Adobe who make sure I always have the latest software; and to Kim Darosett, my editor, who kept everything on track.

Contents at a Glance

Table of Contents

Introduction

Designing professional Web sites is not just about making beautiful pages: It's about understanding your audience and crafting an information structure that not only meets their needs but fulfills business goals as well.

It's about working with a team of people, and understanding the interworkings of the production process from content development through to visual design, comp production, and technical integration.

Over the course of the next 300 or so pages, I show you how to understand the Web design process from start to finish, with an emphasis on creative design and development. At the end of this book, you'll have the understanding it takes to tackle a major, commercial Web site project. You'll still need lots of practice and experience to turn out the good stuff, but this book gives you the solid foundation that you need to succeed.

About This Book

This book is written for both the creative professional who's looking to get into the world of Web design, and the business professional who needs to understand the Web creative and production process in order to manage it. I'm not talking about building personal sites with frilly fonts and loud background patterns. I'm talking about building enterprise-level Web sites for real-world clients — clients ranging from Fortune 100 companies to start-ups that need high-powered Web sites to function as an integral part of a business.

Whether you're managing the process from an executive standpoint, or are a contributing team member, you'll find that the processes, tips, and techniques covered in this book are essential to every project.

By the end of this book, you'll know how to:

- Understand the team roles and responsibilities required to build a Web site
- Present work to clients
- Turn a wish list of content into an information design strategy
- Create wireframe diagrams to plan each unique page layout
- Craft visual design strategies that enhance usability and create a unique brand statement
- Choose software programs for building Web graphics

> ✔ Design graphics that download quickly and look great across platforms and browsers
>
> ✔ Design a user-friendly navigation system for a site
>
> ✔ Organize and conduct user tests
>
> ✔ Make technology choices

Conventions Used in This Book

Throughout this book, I use conventions in the text to make things easier to understand. For example, if I'm introducing a new term, I put it in *italics* and then define it. When I first use an industry-specific term such as *site map,* I make it italic and then give you the scoop on what it means. Usually, these terms are also accompanied by a Web Speak icon. (See the section "Icons Used in This Book," later in this Introduction.)

Code listings and Web addresses are set in a monospaced font like this: `www.dummies.com`. If I want to call your attention to a particular line or section of the code, you'll see it set in bold like this: **`<body>`**.

Foolish Assumptions

This book is aimed at people who suddenly find themselves in the business of designing professional Web sites. Whether you are a project or account manager looking to understand the creative and production process, a business person that needs a Web site, or a programmer that is looking to widen your creative capabilities, this book is for you.

This book is also tremendously helpful for seasoned designers and artists in other fields, such as print design and architecture, who now want to apply their creative talents to Web design. While this book is professional in focus, it is also helpful for those of you who have built personal sites and now want to take them to the next level.

You don't need to know HTML, the coding language of the Web, or high-tech programming languages in order to get the most out of these 18 chapters. In this book, you can find everything you need to know about the people, the planning processes, user interface design, graphic design, and the technologies to start on your journey as a professional Web designer. This book doesn't give you magical creative powers. It does, however, help you channel the creative juices you have into building better-looking, user-friendly, and efficient Web sites.

What You Shouldn't Read

Whatever you do, don't let the technical stuff in this book lead you astray. Throughout this book, and especially in the later chapters, I include some code examples and explain the basics of how they work. As a Web designer, you don't have to be a crack programmer; you just need to be familiar with the underlying technologies and their capabilities. The more you get into Web design, the easier it is to understand the technical stuff, and it won't look as scary.

Whenever you see the Technical Stuff icon in the margin like this, you can choose to turn a blind eye and know that you won't miss out on too much. After all, this book is geared toward creative professionals looking to apply their skills to designing Web sites, not building laser-guided satellites.

How This Book Is Organized

This book is organized to follow the basic workflow of a major Web site design project. Part I starts out with an introduction to the team members involved, and the production process you'll follow. Part II begins the Web production process by first gaining an understanding of the audience and then developing structural plans for your site. In Part III, you'll discover visual design strategies and how to prepare Web-ready graphics. To round out production, Part IV covers the essential techno-babble you need to understand, and finally, Part V sums everything up in a handy reference guide. Whew! Allow me to break it down:

Part 1: The Web Design Kick-Off

Professional Web site design involves a lot of moving, interconnected tasks. To be a successful Web designer or Web manager, you must understand the entire production process and the people you'll work with along the way. Chapter 1 introduces you to the roles and responsibilities of a typical Web project while Chapter 2 outlines the production process and how to manage it.

Part II: User-Friendly Design

Understanding your audience and then crafting a site structure that not only makes sense to them but also attains business goals is a tough balancing act. Chapters 3 and 4 help you to draft the blueprints for your Web site, and Chapter 5 helps you to design visuals that help people successfully navigate your site. Chapter 6 shows you how to test your designs with the end user to see how well they work before you invest a lot of time in final production.

Part III: Designing Web Graphics

Designing the actual graphics for a Web site is the fun part. Chapters 7 through 12 discuss graphic design issues and techniques according to how they relate to the Web, along with all the technical color theory and palette stuff that you need to know. I also show you graphic production techniques and how to prepare client presentations.

Part IV: Producing the Final Web Site

After you determine the graphic and user interface design, the real work begins — assembling the designs into a working Web site. Here's where the scary technical stuff comes in. Don't worry — Chapters 13 and 14 give you a friendly tour of the inner workings of HTML, the basic language of the Web, and an introduction to Cascading Style Sheets, or *CSS*. Chapter 15 takes you a little further and illuminates the technologies that really turn Web sites into movin', groovin' business machines.

Part V: The Part of Tens

True to the *For Dummies* style, Chapters 16 through 18 sum up the contents of the book into Top Ten lists that you can use as handy reference guides. Rip these chapters out and stick them under your desk at work where you can easily access them without anyone ever knowing. Your boss will be impressed with the fountains of knowledge that you suddenly possess. (Then, of course, you'll have to buy a second copy of *Web Design For Dummies* that's undamaged.)

Icons Used in This Book

To make this book user-friendly, I've tagged various sections with icons that point out cool ideas, things to look out for, and industry jargon. As you read, be on the lookout for these little guys:

In talking about Web design, it's impossible to avoid the techno-babble. That's why I like to give you a little advance warning with this icon so you can mentally prepare. The technical stuff is there to give you background, but if it makes your head spin, you can choose to ignore it guilt-free. I won't blame you.

The Web design landscape is littered with land mines that can get you into trouble. Pay special attention to the stuff marked with the little bomb icon.

I love a tasty morsel of advice. I use this icon whenever I've got some cool inside information to share with you.

This icon is not exactly a bomb threat warning, but it does mark things that you should keep in mind during the course of a Web site project.

Like any other industry, Web design is fraught with insider terms. To make sure you get a high-class education here, I've pointed out all the good ones so you can carry on an informed conversation.

I'm Here if You Need Me!

Let me know what you think of the book (good or bad), if you have questions, or if you just have a good design story to share. I'm all ears at `lopuck@lopuck.com`.

You can also contact the publisher or authors of other *For Dummies* books by visiting the Dummies Web site at `www.dummies.com`. The snail-mail address is

Wiley Publishing, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256

Part I
The Web Design Kick-Off

The 5th Wave — By Rich Tennant

"Hold your horses! It takes time to build a home page for someone your size."

*P*rofessional-grade Web site production involves a lot of complex, interdependent tasks and requires an experienced team of people to get everything done. As you embark on your Web design career, a good first step is to understand this production process from start to finish, and the people involved, so that you have a holistic picture and know where you fit in along the way. In fact, many design firms tout their mastery of the Web production process as one of their differentiators when trying to win bids.

In Chapter 1, I introduce you to the different types of roles and responsibilities associated with the Web production process. In Chapter 2, I take you through the whole production process and how design agencies and in-house corporate design teams manage Web projects from start to finish.
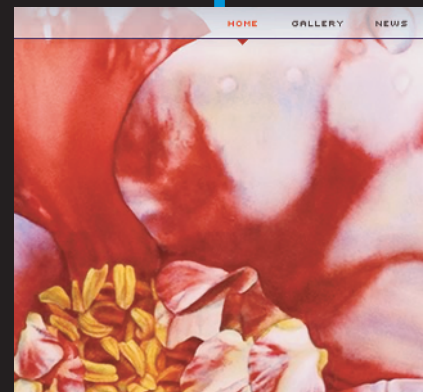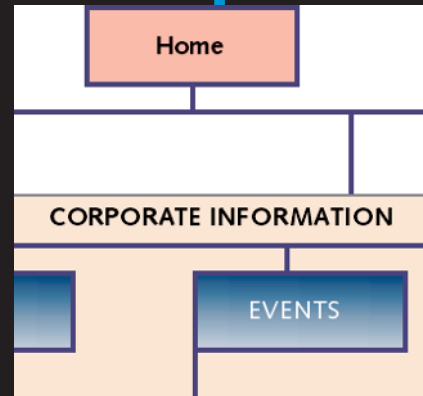
So You Want to Be a Web Designer?

. . . .

In This Chapter

▶ Understanding team member roles and responsibilities

▶ Getting started on your Web design career

. . . .

*T*he Internet industry has been exploding since its inception — from Web sites to wireless, the opportunities are endless. This is great news for you if you're thinking about becoming a professional Web designer. The industry is still relatively new, and a lot of territory remains uncharted. Exciting, rapid developments are around every corner.

Web design is not just about creating a single Web page that looks pretty. In this book, I show you how to design a whole collection of pretty pages that also link together in a way that makes sense to the user. Modern Web sites can consist of hundreds of pages. As a professional Web designer, your job is to know how to integrate design and navigation using the myriad of technologies and techniques to build an effective site.

ASHLEY SINCLAIR

HIGH STYLE EQUINE ATTIRE

designer equine attire

DRESSAGE TRAINING PAD
HUNTER / JUMPER PAD
High-quality printed cotton
Pleather trim
Machine washable

Does this task sound daunting? Never fear, that's what teams are for! People who can do everything from HTML coding to programming to graphic and interaction design are rare. If you want to be a Web designer, you simply need to understand enough about the entire process from start to finish, and the role that every team member plays, to enable you to focus on the fun stuff — design.

In this chapter, I introduce you to the different players that you'll surround yourself with on your journey through professional Web design.

The People Involved

Designing Web sites is such a huge undertaking that to do it right, you really need a team of people. Here is a sampling of the major players, their roles, and when you need 'em.

Business folks

In the early days, you could get away with sticking a Web site up on the Internet and expect to get reasonable traffic without much further effort. In the crowded Internet highways of today, however, you really need a business strategy and a marketing plan. The business folks must be involved with the Web site from the very beginning. They are in charge of the following responsibilities:

- ✔ **Setting the goals for the site.** You must always understand the business goals of the site and the profile of the end user so you can craft a Web site that fits the customer's needs and achieves business objectives.

- ✔ **Reeling in the visitors.** The marketing folks' biggest task is to figure out how to steer Web surfers to your site. In the Internet business, *getting eyeballs* (fun industry jargon for getting people to look at your site) is not as easy as it sounds and involves search engine optimization (SEO), partnering with other companies, and an integrated online and offline campaign strategy. For these reasons, marketing folks need to get crackin' on their plan right away.

Producer folks

After you get clients excited about a Web project, their eyes tend to get bigger than the budget. Among many other responsibilities, the producer's job is to set and manage client expectations so the project stays on track. Using a variety of tools such as Microsoft Project, shown in Figure 1-1, the producer must keep the project, the team members, the client, and the budget on track from start to finish.

One of the most common problems a producer must address on a project is *scope creep*. Features and functions that you did not plan for have an uncanny ability to find their way into the design. Either project team members are trying to prove themselves by gold-plating their contributions, or clients are scrutinizing the site and suggesting way too many changes. In either case, constant noodling can add up to more time and money than you had planned, so either plan for it, or catch it and stop it before it happens.

	Task Name	Cost	Duration	Start	Finish
ⓘ	□ **Design Conference Web Site**	**$447,554.55**	**75 days?**	**Mon 8/1/05**	**Fri 11/11/05**
	□ **OVERHEAD**	**$274,859.10**	**75 days**	**Mon 8/1/05**	**Fri 11/11/05**
📋	Project Team Oversight	$124,859.10	75 days	Mon 8/1/05	Fri 11/11/05
📋	Contingency	$40,000.00	0 days	Mon 8/1/05	Mon 8/1/05
	⊞ **External costs**	$110,000.00	0 days	Mon 8/1/05	Mon 8/1/05
	⊞ **DISCOVERY PHASE**	**$0.00**	**27 days**	**Mon 8/1/05**	**Tue 9/6/05**
	Discovery complete	**$0.00**	**0 days**	**Mon 8/1/05**	**Mon 8/1/05**
	□ **PRELIMINARY DESIGN PHASE**	**$172,695.45**	**75 days**	**Mon 8/1/05**	**Fri 11/11/05**
📋	Team resources	$172,695.45	75 days	Mon 8/1/05	Fri 11/11/05
	⊞ **Internal project kick off**	**$0.00**	**1 day**	**Mon 8/1/05**	**Mon 8/1/05**
📋	**Internal kick off meeting**	**$0.00**	**1 day**	**Tue 8/2/05**	**Tue 8/2/05**
	⊞ **Preliminary wireframing**	**$0.00**	**8 days**	**Wed 8/3/05**	**Fri 8/12/05**
	Preliminary wireframes complete	**$0.00**	**0 days**	**Fri 8/12/05**	**Fri 8/12/05**

Figure 1-1: Most Web producers use project tracking software like Microsoft Project to manage schedules, resources, and milestones.

Visual designers

The Visual Designer works closely with the team to craft not only the site's structure, but also the navigation and user interface design. Additionally, the designer is in charge of the appearance of the site — integrating text, graphics, and animation to create a unique look that suits the client's goals and branding, yet is easy to use and practical to implement technically.

I find that many Web designers are print design expatriates. If you're transitioning from the print design world, the hardest thing you need to learn is how to maximize the technologies and navigation options at your fingertips to design effective Web interfaces.

Project management

Kelly Goto

Principal, www.gotomedia.com

Successful project management is keeping the members of the Web development team "on the same page" throughout the project. Balancing the needs of the client, the goals of the site, and the reality of scope and budget are challenging tasks. Establishing clear communication means understanding the needs of the client and individual team members. Following a process and understanding the overall goals and objectives of the site from the onset is also critical to the success of a project. The goal is to maintain clear objectives through each phase of development, to manage *scope creep* (the tendency of projects to expand in size), and to predict the future.

Many print designers create graphically heavy, magazine-like interfaces that certainly look cool, but are not very practical for the Web. They download slowly and are hard to automate or update. The Web page in Figure 1-2 is 8½ x 11 inches and has huge graphics. This page has navigational choices at the bottom, but they aren't visible because the page is larger than the browser window, which requires the user to scroll to see them. Compare this to the design in Figure 1-3. In Figure 1-3, the main navigation is clearly set apart and is high on the page. To be an effective Web designer, you must understand how Web pages are built so you can maximize usability and page efficiency.

Figure 1-2: This design is graphic-heavy, the navigation is not obvious, and it is difficult to build and update in HTML.
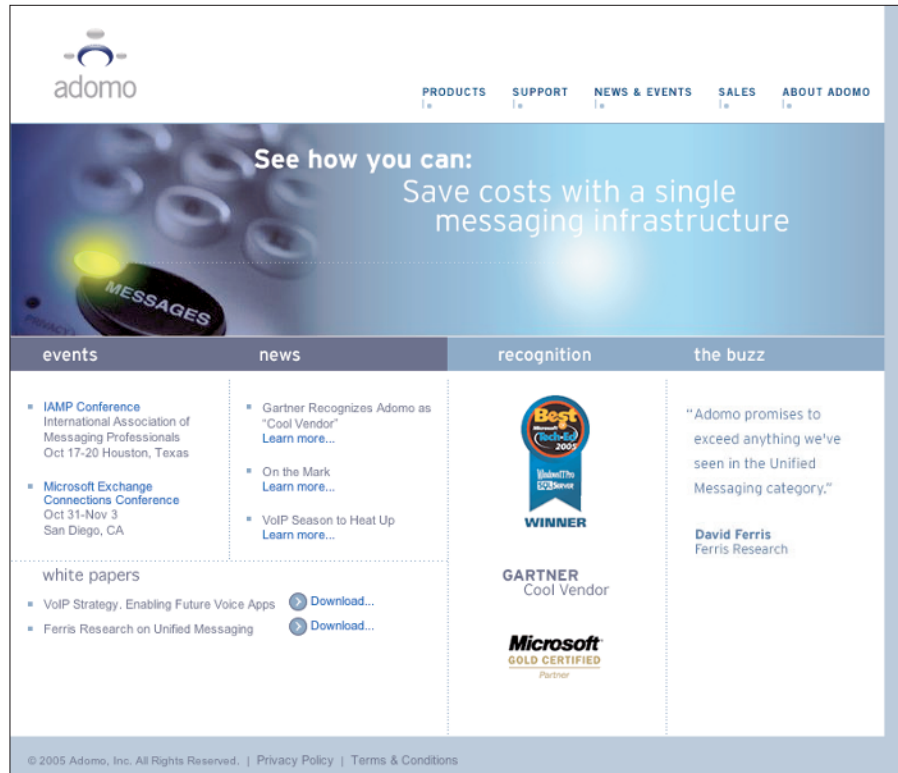
© Adomo, Inc. www.adomo.com

Figure 1-3: This modular design is easy to create and update in HTML, and it clearly outlines the navigation options for the user.

Information architects

This impressive-sounding title goes to the person whose job it is to sit down and figure out how the whole site fits together and how people will navigate from one page to the next.

WEB SPEAK

One of the first tasks of an Information Architect is to design a *site map* diagram, like the one shown in Figure 1-4, that shows all the main sections of the site. The IA, as this person is often referred to, then dives into the page-level detail and creates a series of *wireframe* diagrams, like the example shown in Figure 1-5, that show the content and navigational elements that go on each major page of the site. Between the site map and the wireframe diagrams drawn for each page, the Information Architect, in effect, builds the blueprints for the entire site.
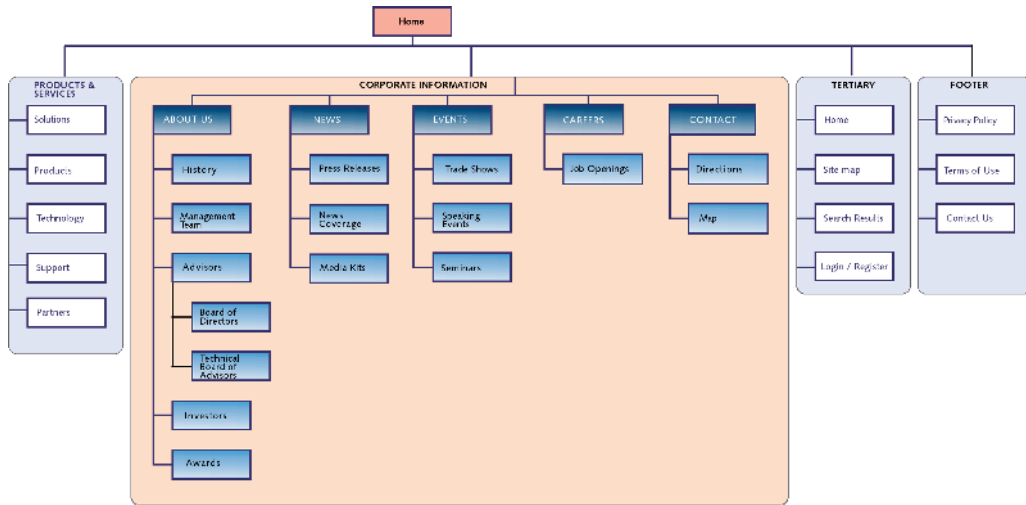
Figure 1-4: A site map is like a bird's eye view of your Web site showing all its sections.
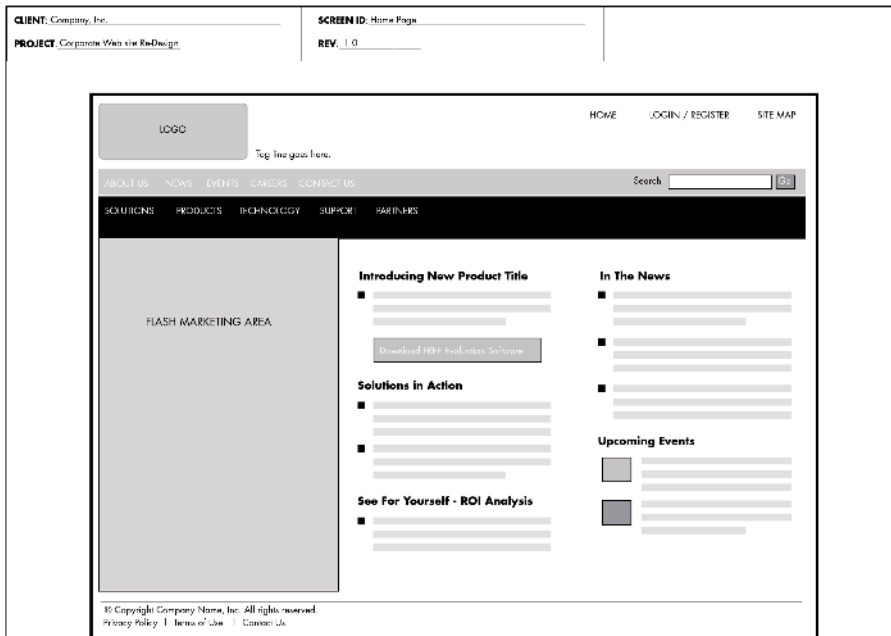
Figure 1-5: A wireframe shows the structural layout and content plan for a page.

Content designers

The content team is in charge of all the text on a Web site. These days, some Web sites have as much copy (a.k.a text) as a magazine or a book, and like a magazine, must be designed in such a way that they can be updated on a regular basis without reinventing the page layout each month.

For this reason, a Web project often has two roles:

✔ **Content Strategist:** The person who identifies the chunks of copy needed for each page — for example, headlines, bullet lists, and descriptions — and the rules for each, such as character limits and word counts.

✔ **Copy writer:** The person who actually writes the text for each chunk.

Writing for the Web is a whole other animal. Firstly, no one likes to read text-heavy Web pages, so the Content Strategist and the writer have to convey the most impact in the least amount of space. Secondly, you need to lead off with the real meat of the message — the conclusion first and then follow with a few supporting details in case people get that far.

REMEMBER

You have, at most, about three to four seconds to get your main message across with words and graphics to hook your visitor. If visitors don't get it, they're off to the competition. After all, it's just as convenient to type their URL as it is to type yours.

Media specialists

No modern Web site would be complete without a splash of video, Flash, or audio media. With so many specialized media formats and compression schemes, however, it's best to leave media design to separate professionals. This is especially true for Flash development. Flash is a software application that can create highly interactive, game-like applications and animation (look for Flash at www.macromedia.com). The program is so powerful that Flash design has become its own highly sought after profession.

HTML slingers

A variety of titles — none of them standard — denote the people who assemble Web pages in *HTML* (HyperText Markup Language). "Technical integrators," "technical producers," and a few other titles come to mind, but the most amusing title I heard while working at eBay was HTML Slinger. This funny title made sense because the folks who specialize in this stuff are creative programmers who have hands that can type HTML faster than you can say "draw."

Although HTML is a coding language, being able to use it well involves a lot of creativity. For one reason, different browsers interpret code differently, which can really screw up your page layout. Good HTML slingers are great at finding workarounds for these browser incompatibilities — all the while maximizing a page's download performance.

Programmers

WEB SPEAK

Modern Web site projects would not be complete without a team of programmers. These folks can really give your Web site a turbo boost by making it dynamic. (In the industry, the word *dynamic* refers to an automated Web site that connects to databases for displaying and storing content and/or connects to applications to process input.)

A Web site often consists of hundreds of pages — well, that's only a half-truth. Programmers can help you build a few template Web pages that you can use again and again throughout your Web site. This way, you can quickly create a Web site consisting of hundreds of pages, as shown in Figure 1-6. An online database *populates* (industry term for *fills in*) the template with different information to create each new page.

WEB PAGE

TEMPLATE "A"   TEMPLATE "B"   TEMPLATE "C"

ONLINE DATABASE
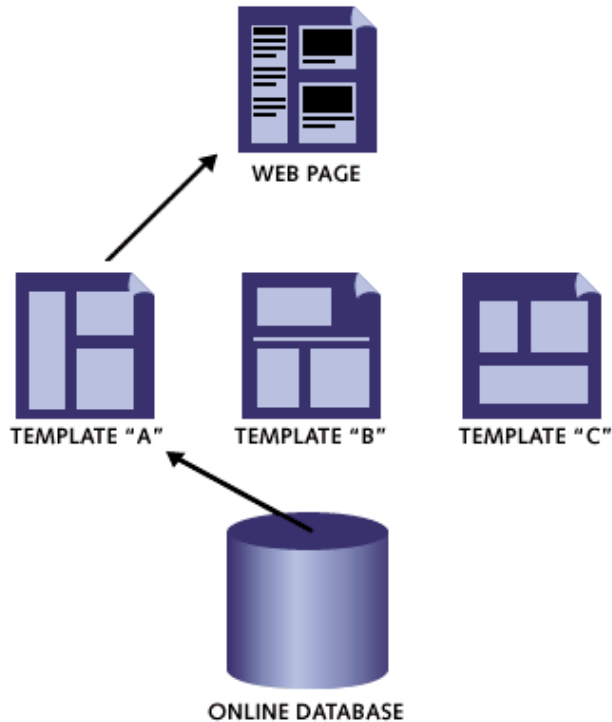
Figure 1-6: A few HTML template pages and a database can create hundreds of Web pages.

In addition to coding the template Web pages with techie software development packages like the scripting language PHP (*Hypertext Preprocessor* — go figure), ASP *(Active Server Pages),* and ColdFusion, programmers also create the online databases that house all the information. Building databases can

be so complex that many times you need a specialized database dude or dudette for that task alone!

Getting Started in Web Design

Now that you have a feel for the different types of professionals you'll be working with — or become yourself — it's time to start taking steps toward your own Web future (or interactive TV, mobile design, DVD) by educating yourself on the production process.

The rest of this book dives into the nitty-gritty of the Web design workflow so that you can understand how a site is built from start to finish. The following are the kinds of questions you need to answer before you embark on your career: Where do you begin building a site? How do you organize information? How do you design a site that's easy to navigate? How do you design and build graphics? And finally, how do you build a site in HTML and what do all those technical acronyms stand for?

Getting Experience

The classic chicken and the egg question comes to mind yet again: How do I get experience without having a job, and how do I get a job without having experience? The answer I think is to go out and create your own experience and then present it in a nice portfolio ready to show to a potential client. Here are a few ideas for you to consider:

- **Help out a friend.** All of us have friends who need a Web site designed for some legitimate purpose. First read *Web Design For Dummies* cover to cover and then offer to design a simple site for a friend that can showcase your strengths. No one will ever know that Suzie's Seafood was a site you designed for free for your best friend.

- **Partner with other Web professionals.** After working on Suzie's Seafood, you'll realize that building a Web site is not a one-person job. Now that you have a taste, try to expand the complexity of your next site by partnering with one or more Web professionals. Again, offer your talents free of charge.

- **Create a portfolio.** Design and build your own Web site. Make it a showpiece that exhibits your work in the best light. You may need to call on some favors from other Web professionals to help make it what you want it to be.

- **Learn from other sites.** The Web is the best place to get new design ideas. Scour the Site of the Week listings at www.designinteract.com. Not only can you get a lot of visual and interaction design ideas, you can often get HTML and JavaScript code samples by viewing the source.

When you feel confident you've developed enough skill level to take on a paying customer, start looking online for different opportunities. Craig's List (`www.craigslist.com`) is a fantastic resource for finding local freelance design work.

Designing an award-winning site takes years of knowledge coming together. Not to state the obvious, but sell yourself appropriate to your skill level and comfort zone. I have heard many clients groan about inexperienced Web designers who got in over their heads and could not deliver. Better to start small and build up experience and a solid reputation.

From Concept to Execution

In This Chapter

▶ Determining site goals and requirements

▶ Creating a project plan

▶ Building site maps and wireframes

▶ Developing your content and visual design strategy

▶ Testing, launch, and maintenance

Developing a professional-grade Web site is a big undertaking that is akin to producing a movie. With so many moving parts, people to wrangle, and steps involved — from compressing graphics to tracking bugs and licensing audio clips — you need to approach the madness with some sort of methodology.

A lot of fancy Web design companies like to refer new clients to their patent-pending, proprietary "five-step design process" to educate them on the chronology of the development process. The five steps are Definition, Design, Development, Testing and Launch, and Maintenance. The process isn't ultra-special or top-secret, it's just advanced common sense nicely spelled out for the client.

In this chapter, I take you through the process that most Web agencies and internal corporate development teams follow. Incidentally, the process is pretty much the same for DVD and mobile content development, so this is a must-read chapter.
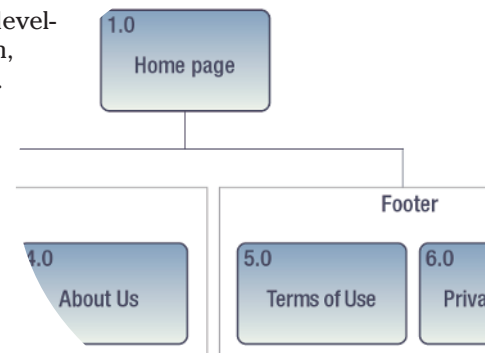
Phase 1: Definition

As tempted as you may be to jump in and sketch out a new site's structure, Phase 1 should be a "discovery" period where you define the business goals and understand the customer needs you're designing for.

Understanding site goals

From small family businesses to companies the size of Disney, step one is the same: Understand, in priority order, the top three or so goals that the company is looking to achieve with the design or redesign of the site. These goals directly influence the site's structure, its visual design, and the layout and content choices you need to make for each page.

For example, when redesigning a site for a swim school, the client expressed the need to convey multiple locations, provide quick access to key information for current customers, and to encourage prospective clients to call its phone number so a sales agent could customize a program and personalize the experience. La Petite Baleen's home page, shown in Figure 2-1, features content elements and a design that supports all three of its business goals.

Gathering business requirements

The next useful exercise is to sit down with the client and make a list of all the things the site needs to do. Here's a tip: Think in terms of finishing a bunch of these sentences "The site should have the ability to X."

Here are some guidelines:

- ✔ **Keep it high level.** Lay ground rules up front with the client for this exercise. In my experience, clients can quickly dive into *how* things should be done instead of focusing on *what* needs to be done.

- ✔ **Make a list.** Capture a bunch of "ability to" sentences like:

  - The ability to register

  - The ability to log in/log out

  - The ability to showcase a featured product each month

  - The ability to allow our team to update home page content each day

- ✔ **Prioritize.** Group each of the "ability to" statements by priority such as *1* is a must-have, *2* is a nice-to-have, and *3* is a could-have-in-the-future.

Building a project plan

After you have a good feel for the level of work involved in the new site, you can assemble a project plan that lists all the steps — organized by these patented five phases — and figure out how much time, money, and people you need to accomplish each step.

TIP

An easy way to start a project plan is to work backwards from the launch date and sketch out some major milestones. Doing so can give you a better perspective of the project's *pacing* as I like to call it.

Multiple locations are included in header

Call to action is visible   Key information is surfaced on home page

© La Petite Baleen. www.swimlpb.com

Figure 2-1: The first step in Web design is to understand the business goals of the site, which influence the site structure, content, and design choices.

Most Web producers use Microsoft Project to map out all the steps, assign team members to tasks, and set up *dependencies* between tasks — or, in other words, things that rely on other things happening first. Notice the stair-stepped *gantt* view of the visual design steps in Figure 2-2. The designer first creates a few design directions, the client then chooses one for refinement, and finally the look and feel is established. If the client takes longer than two days to choose a design or rejects the designs presented, the schedule slips, and the launch day is in jeopardy.

⊟ **Look and feel development**	$0.00	13 days	Wed 8/3/05	Fri 8/19/05	
Collaborate on preliminary wireframing	$0.00	6 days	Wed 8/3/05	Wed 8/10/05	
Develop design directions	$0.00	5 days	Thu 8/11/05	Wed 8/17/05	
Review with client	$0.00	1 day	Mon 8/15/05	Mon 8/15/05	
Refine design from client feedback	$0.00	3 days	Tue 8/16/05	Thu 8/18/05	
Client sign off on design direction	$0.00	1 day	Fri 8/19/05	Fri 8/19/05	
Look and feel complete	**$0.00**	**0 days**	**Fri 8/19/05**	**Fri 8/19/05**	8/19

Figure 2-2: In the *gantt* view, each project task is represented as a bar. Notice how some tasks must be completed before others can begin.

Phase 2: Design

When you buy or receive toys labeled "some assembly required," you know you're in for a brain-twisting treat. If you're like me, you spend the first half-hour attempting the task without the aid of instructions. Not until you pull some hair out do you finally surrender to reading the manual. The design phase of a Web site is your chance to write the instruction manual for the team before anyone begins pushing pixels around.

Creating the blueprint for the site

WEB SPEAK

A primary task in the design phase for any Web site, large or small, is to create a blueprint, or *site map* as it's called in the industry. Without a site map like the one shown in Figure 2-3, you're headed for frustration. For one thing, the team won't have any unified direction. Secondly, without a plan, you can't possibly anticipate all the content, pages, and features that need to go into the site.

For example, imagine getting halfway through the design and then realizing that you forgot to include a critical feature like a resources area. Now you find yourself redoing an interface that could have been done right the first time — if you'd only had a plan. Check out Chapter 3 for the lowdown on how to create a site map.
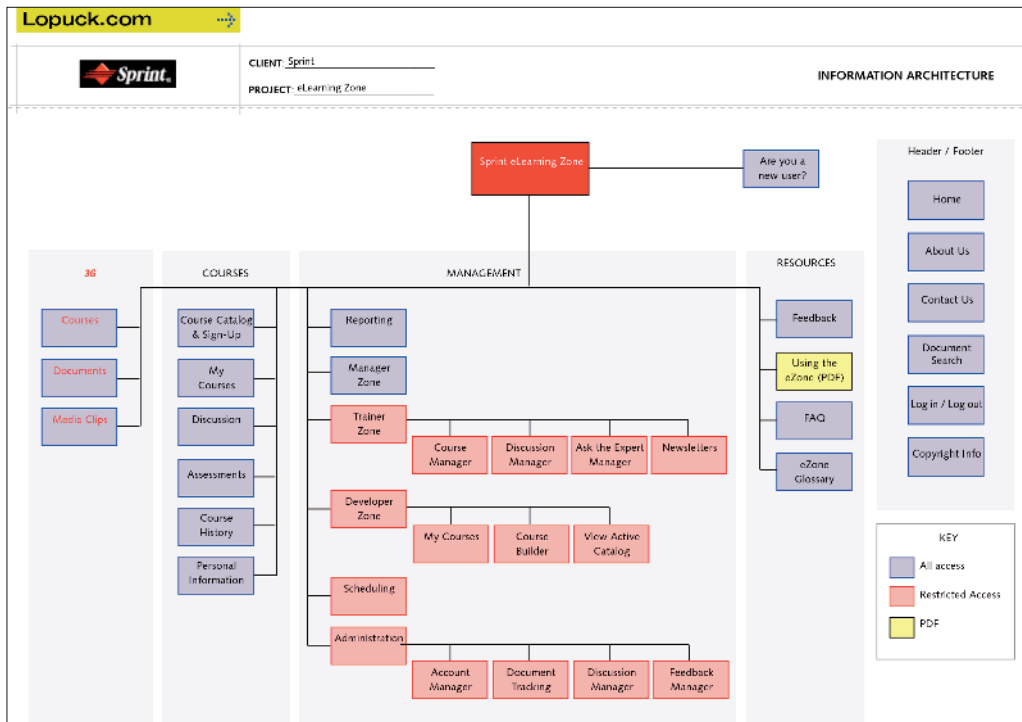
© Sprint, Inc.

Figure 2-3: A site map shows all the major sections of a Web site and how they interconnect. The interconnections lay the groundwork for your navigation scheme.

Designing a plan for each page

While the site map shows the Web site from a bird's-eye view, it doesn't give you the detail you need to design and build each page. For that, you need a series of *wireframes*.

During the Design phase, the Information Architect collects input from all team members and builds a diagram layout, called a *wireframe,* for each major page of the site. A wireframe like the one in Figure 2-4 shows the following elements:

- *Global navigation* scheme (the navigation that appears on each page of your site)
- Text and media chunks and their relative placement
- Interaction design (how people use the elements on the page)

See Chapter 3 for details on building wireframes.

Company Logo

STORE TITLE
Store description lorem ipsum consectetuer dulce ets

| Home | Products | Services | About | Account | Contact |

Categories
> Category Link
> Category Link
> Category Link
> Category Link
> Category Link
> Category Link

Lorem ipsum dolor sit amet, consectetuer dulce et decorum est e pluribus unum fiat lux semper fidelis ad infinitum. Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consectetuer dulce et decorum est. **LINK>**

IMAGE

Login
Username

Password

LOGIN

Feature Headline

Lorem ipsum dolor sit amet, consectetuer dulce et decorum est e pluribus unum fiat lux semper fidelis ad infinitum. **LINK>**

100 x 125

Feature Headline

Lorem ipsum dolor sit amet, consectetuer dulce et decorum est e pluribus unum fiat lux semper fidelis ad infinitum. **LINK>**

100 x 125

©2005 Company Logo. All Rights Reserved | 1234 Main Street, Any City, USA | (123) 555-1234

Figure 2-4: A wireframe is a diagram-like plan for a Web page.

Implementing user testing

Testing wireframes with users is an iterative process that you start in the Design phase and continue well into the Development phase. Testing early in the Design phase is a good way to see if you are on track because you can quickly make adjustments before going too far down the production path.

In Chapter 6, I show you techniques for testing important features of your Web site by building a series of *clickable* wireframes and sharing them with a group of users that represent a good cross section of your target audience.

Putting together a content plan

Hand-in-hand with wireframe development is figuring out a content plan. After all, how can you suggest content like a news area on a page without knowing where the news will come from, how extensive it should be, and how often the client will be able to update it?

If the site will be using a *content management system* (a database system that houses all your text and feeds it to your pages), the Content Strategist creates a list of all the kinds of text components that go on each page and categorizes them into types like headlines, captions, and descriptions and subtypes within each type. A Content Strategist also establishes rules for each content type, such as how many text characters it can contain.

Lastly, during the Design phase, the content team thinks about the writing tone that would be best for the site given the business goals established during the Definition phase. For example, should the site's text read warm and inviting, or high-energy and exciting, or as just-the-facts? The writers need this input to guide their writing style during the Development phase.

Establishing "look-n-feel"

After you finalize the site map with the client and develop the wireframes to the point where content and interactive elements contained on each page are more or less decided, the visual designers can explore different design directions for the site.

The general practice is to create a design treatment for the home page and a *subpage* (anything but the home page). This forces the designers to figure out a design that can tie the whole site together and account for the site's navigation scheme. I like to have at least three different design directions developed by three different designers. This strategy gives you enough variety to give the client some options. Figure 2-5 shows two designs made by two different designers from the same wireframe shown in Figure 2-4. Notice how each design conveys a very different feel or mood.

Getting input from a focus group

Most often the client is the one who chooses the design direction — usually with one or two rounds of noodling or "frankensteining" elements from different designs together. For high-profile Web sites, however, it's a good idea to get input from a focus group to see which design resonates most clearly with them before you present to the client. That way, you have customer feedback ammo when you present and can help steer the client in the right direction.
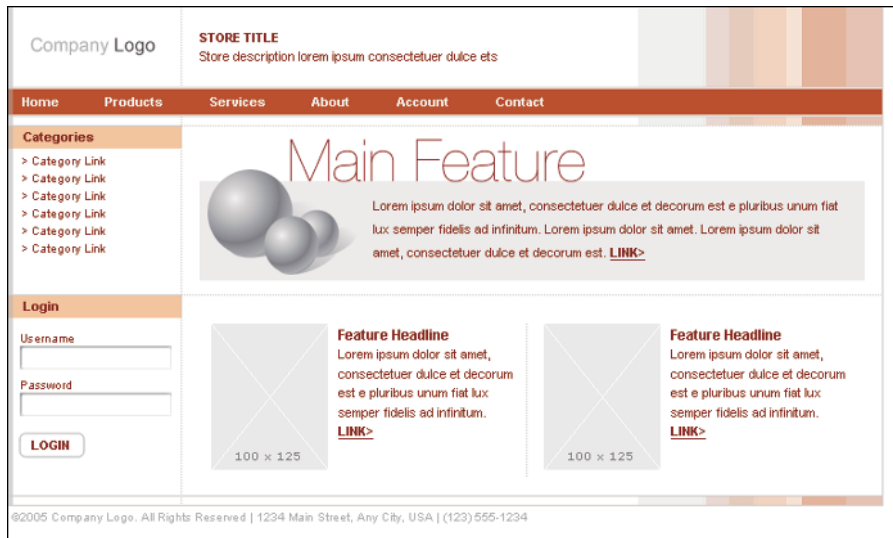
Figure 2-5: You can derive two very different designs from the same wireframe.

Phase 3: Development

The time has come to commit pixels to the screen and move on to the Development phase. In this phase, all video, audio, Flash animation, and graphics are prepared for the Web: They're compressed to speed up their Web travel, saved in a Web-friendly format, named, and uploaded to the server. All programming and HTML work is done, and the site comes together, ready to be tested.

Producing final Web graphics

After the client chooses a design direction for the site, the grunt work can really begin. The most efficient way to produce the multitude of necessary graphics is to work from design templates. For example, most Web sites have a number of elements that are frequently repeated, such as buttons, headlines, and navigation bars. Using design templates ensures that your site doesn't appear to have multiple personalities and enables you to delegate graphic production to a team of people while maintaining design consistency.

Use a program such as Adobe Photoshop or Macromedia Fireworks to produce not only the template designs but also the final images for the site. Notice all the layers in the Photoshop file shown in Figure 2-6. From this one "button template" file, you can export all the buttons needed for a site.
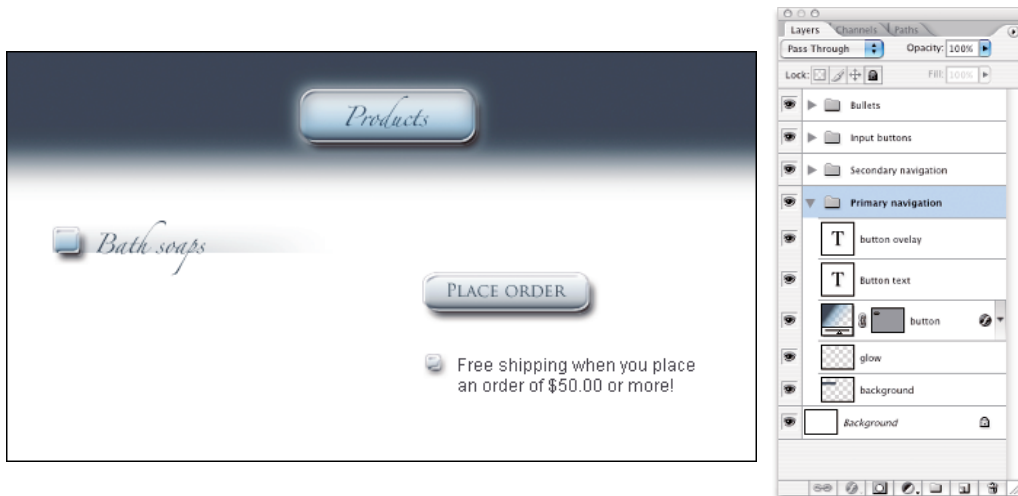
Figure 2-6: One Photoshop file can have the layers needed to create and export all the buttons needed for a Web site.

Content development

Meanwhile, the content team is busy writing all the *copy* (text) components for the site. How do they know what to write? If you recall in the Design phase, the Content Strategist maps out all the types of copy needed for the site, how many copy blocks of each type are needed, and on which pages they live. Content strategists, along with the writers, also decide what the tone of the site should be.

Armed with a checklist of copy to write and the style to convey, the writers can begin their task.

Media development

These days, most sites have some sort of rich media — whether it is Flash animations and applications, audio, or video. In fact, some sites like the one shown in Figure 2-7, are entirely built in Flash except for the one HTML page it sits on.
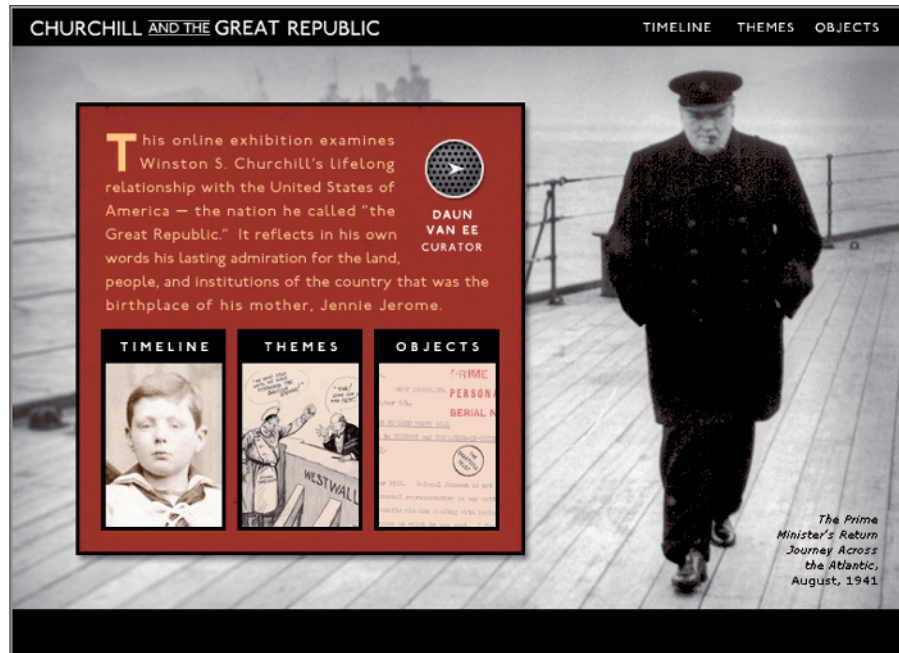
Courtesy of The Library of Congress. www.loc.gov/exhibits/churchill/interactive/

Figure 2-7: All the interaction for this site is contained inside a Flash movie on a single HTML page.

Flash development can be so complex that it has become a process and a profession unto itself. Not only do Flash designers need to create their own version of a site map and wireframes to make their plan of attack, but they also need to be one part designer and one part gear head in order to craft an effective Flash element.

As for audio and video production, both involve acquiring source material to work with. This means that you either need to license existing material or, you guessed it, go out and get it yourself — which is a whole other fun time. You need locations, casting, talent wrangling, permissions, release forms, wardrobe, and multiple takes to plan for and manage before you can get to the editing bay. Even after the editing sessions, you then need to compress the files and save them in a Web-ready format. As you can imagine, depending on the project, media can be a hefty part of your budget and timeline.

Holding it all together with HTML

Like the joke that says, "Duct tape holds the world together," HTML is certainly the glue that holds the World Wide Web together. *HTML* (HyperText Markup Language) is a simple coding language that tells your Web browser how to string formatted text, graphics, tables, *CSS* (yet another acronym that stands for Cascading Style Sheets), and media elements together on the page.

Every page of every Web site, such as the one in Figure 2-8, is built with HTML — even if the HTML is simply organizing some other programming language. You can write HTML code in any simple text editor or use a more full featured text editor such as BBEdit. After you finish, you can load it into a Web browser and see the results.

HTML is pretty limited in what it can do. For example, you can't build an online shopping system with just raw HTML. For that, you need to bring out the big guns — those other programming languages, which I discuss in Chapter 15.

Databases, programming, and things to make your head spin

If you're designing a Web site that's going to do anything beyond a simple show of nice text and graphics, you have to enlist the programmers. Web sites that show personalized messages, allow you to buy products with credit cards, or register for special events all require specialized programming. You can't build this sort of functionality with HTML alone.

Development software, such as ColdFusion, and programming languages, such as Microsoft's Active Server Pages (ASP), are actually integrated right into the HTML code for the page — making it a funky hybrid of coding languages. The HTML portion controls the page layout, whereas the programming

language does all the cool stuff — such as linking to an online database to automatically display a product of the day. Figure 2-9, for example, shows how the ASP code connects to an online database, where it can grab cool stuff like someone's name, weight, and annual salary. The HTML code then neatly formats the information for all to see.

Figure 2-8: Like a coin, every Web page has two sides: the pretty exterior and the HTML code that makes it work.

Figure 2-9: The functions on this page are too complex to do with just HTML.

To build a page like this, you can either roll up your sleeves and type ASP code directly into the HTML page, or you can use HTML tools like Dreamweaver and Dreamweaver extensions. You can find Dreamweaver extensions (shown in Figure 2-10) from WebAssist at webassist.com to give you a power lift.
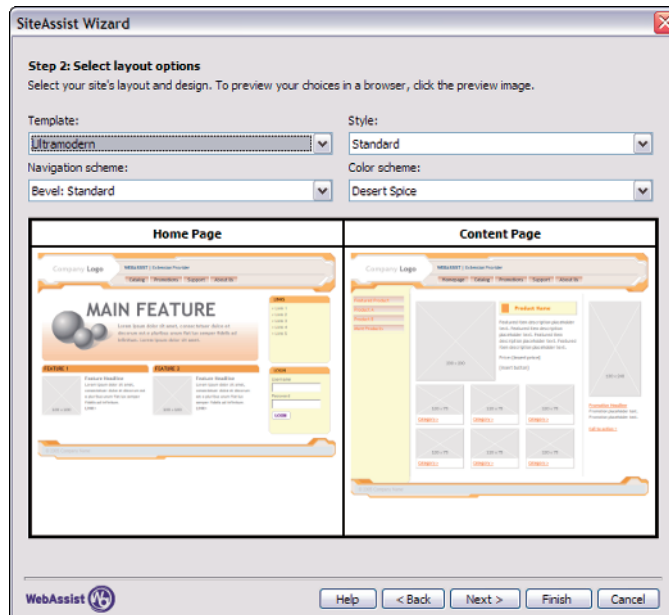
Figure 2-10: WebAssist provides wizard-like tools that help you build complex Web sites in a fraction of the time.

So now that your head is spinning with all the technical to-dos, it's time to start working on the home stretch — testing and launching.

Phase 4: Testing and Launch

REMEMBER

Just because the Web is a flexible publishing medium, don't be tempted to throw the site online and make on-the-fly changes to it as you find problems. Users notice misspellings, broken links, and whatever else and will be left with a negative impression of your business. Before you go live with a site, make sure you subject it to a rigorous testing and *debugging* (correcting errors) cycle.

Quality assurance

So many details go into building a Web site that you're bound to make mistakes along the way. Plan ample time at the end of a project for a formal *QA* (quality assurance) period. For large Fortune 100 projects, I've used every bit of four weeks to identify and correct problems — and often that's not enough time and we have to keep a list of things to fix after launch.

In addition to checking for misspellings, missing images, bad links, and so on, check your site on different platforms (Mac and Windows) and different browsers (Mozilla and Internet Explorer) to make sure that everything works the way you expect it to. The other thing to check is how fast various pages load when viewed through different connection speeds. Rich media, such as Flash animations, video, and audio, or improperly compressed images can really make your pages limp along even on fast connections.

Notice how giving up a slight amount of image quality in Figure 2-11 gains tremendous file size savings, resulting in a faster load time that is a better "customer experience" as they say in Web design circles.

Launch day

During development and testing, you don't want the outside world taking a sneak peek at your site. So, you can hide sites in many ways leading up to launch day — putting them on different servers, enabling *IP blockers* (software that allows only certain computers to access the server), or simply developing sites *locally* (on your computer). Ergo, on launch day, you need to take steps to undo all this hiding to make the site available for all to see at your prescribed moment.

100% quality   65% quality

JPEG (Document, f↴   100 quality
283.14K
45 sec @56kbps

JPEG (Document, f↴   65 quality
45.77K
7 sec @56kbps
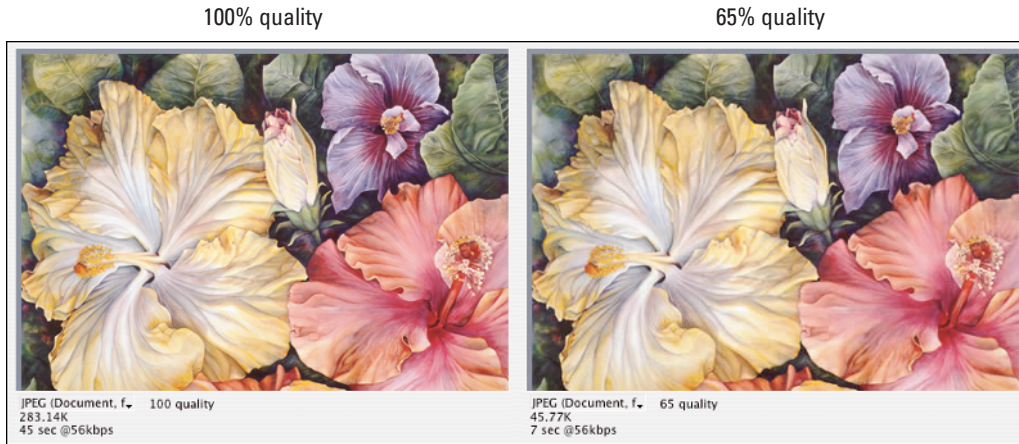
Figure 2-11: By using 65% JPEG compression on the images, the image loads in 7 seconds as opposed to 45 seconds on a 56K connection.

To do so, you need to create a plan that chronicles the steps you and your team need to take leading up to launch time. Things such as letting new URLs (Web addresses such as www.companyname.com) propagate throughout the Internet can take a few days and should be part of your launch plan. Removing IP blockers and setting up *redirects* (sending people to different URLs when they type a URL) are tasks and double-checks that you need to include in the launch plan.

Phase 5: Maintenance

The minute after the site launches marks the beginning of a critical maintenance phase. Once a site is working in the real world on the servers it was designed for, a whole new set of technical problems can arise. I like to watch a site closely for the first couple weeks after it launches to make sure it settles nicely into its new home on the Internet. Keep the engineers close at hand just in case!

Additionally, you need to keep the site content fresh — changing images and text on a regular basis — and continue to fix the bugs left over from the QA period. You can now consider some features that you were not able to include for time or budget reasons for development and add them to the site as upgrades. Larger upgrades or additions are mini Web projects in themselves, in which case you get to repeat these five phases all over again.

Part II
User-Friendly Design

The 5th Wave — By Rich Tennant

SITE MAP STRATEGY BY JERRY
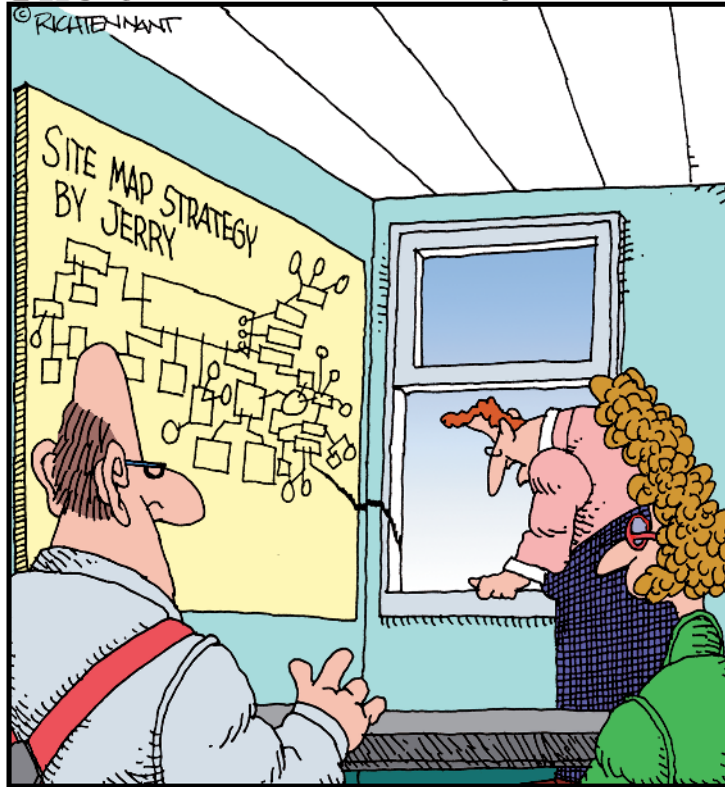
"Okay, well, I think we all get the gist of where Jerry was going with the site map."

*M*ost of us grew up with the mantra "measure twice, cut once" drilled into our brains — and for good reason. A little bit of planning in the beginning goes a long way in saving you a lot of time, effort, and money later in your Web design process.

Just like you'd draft a blueprint before you begin to build a structure, the first step in Web design is to organize all the site content into logical categories and subcategories, a process called *information design*. After you have the site's structure worked out, you must dive into the page-level detail and create diagram-like plans called *wireframes* for each unique page layout. You then need to test all this pre-planning with end-users to make sure it makes sense to them.

Because end-users are the final judge of a good site structure and design, Chapter 3 explores how to identify your target audience and then go about crafting your information design. Chapters 4 and 5 share techniques for optimizing usability. Finally, Chapter 6 shows you how to organize and conduct user tests to make sure your designs are on target.
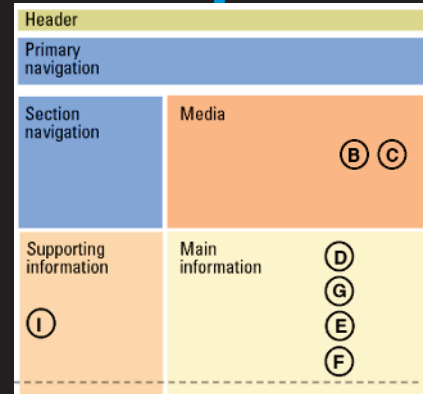
3

Designing the Right Site for the Right Crowd

In This Chapter

▶ Determining your audience

▶ Creating an outline for your site

▶ Transitioning from outline to site map

▶ Building a site map

▶ Deconstructing an existing site

▶ Marketing to your audience

*O*ne of the key balancing acts you as a Web designer must do is reconcile what clients think they want and what their customers actually need. During the Definition stage, you gather a bunch of *business requirements*. These requirements are really just a client wish list because not all the requirements are feasible within the project's timeline and budget, nor are they appropriate for the end user. This is why it's so important to not only define the three or so overarching business goals for the site, but also determine the top three or so *user* goals. These combined high level goals become your measuring stick when determining what kind of content and features to include in the site and how best to organize them.

In this chapter, you get to play the role of *Information Architect.* You determine what kind of people are going to be using the site and think through what kind of things they'll want to do. You find out how to balance customer needs with the client's wish list of requirements and develop an outline and a site map that becomes the foundation of the Web site.

Who Is the Audience?

Before you can design a site, you must first understand the target audience who will be using it. Suppose that you're designing a Web site for a retirement home. If you're not familiar with the assisted-living industry, you may mistakenly assume that the audience for the site is elderly people.

Unless you sit down with your client and ask a lot of questions, you'll never know that the real audience for the site is people in their forties — the people who make plans for their aging parents. This changes everything about your ideas for the Web site — from the graphical appearance to the content features and site organization.

REMEMBER

This example illustrates an important point: Having structured conversations with your clients and asking them many questions about their industry, competition, and intended audience is a critical step in the design process.

Question checklist for clients

TIP

The best way to extract customer profile information from your clients is to have a chat with them and run through a list of prepared questions. What sorts of questions should you ask? Table 3-1 is a generic list to get you started. You should ultimately create your own standard list and modify it for each new project.

Table 3-1	Customer Questionnaire
Question	*Reason for Asking*
How many different types of customers do you have?	Some businesses have more than one type of customer. For instance, they may categorize customers by *current customer* and *prospective customer* or *business customer* and *consumer customer.*
How would you describe each customer type?	Have the client describe this customer in detail. Is the customer male or female? Is he or she a professional? If so, in what field? What is their income bracket? Are they Web savvy? Their education?
What are the barriers for each customer?	Customers always have at least one thing that will make them hesitant to engage in your client's offerings. Is the product perceived as too expensive? If so, your content has to really push the value and/or financing options.
What's the biggest value this new Web site will provide to customers?	You need to know what the Web site will offer that solves a customer's need. For example, will the new site offer exclusive discounts or help customers zero in on the right product for them and compare product features side by side?

Personas

WEB SPEAK

The client checklist can give you a good idea of the customers you're after, but sometimes going a step further and developing a one-sheet profile for each customer type is helpful. In the Web industry, these one-sheets are called *personas*.

Personas have a name (fake of course) and even a picture along with a short bio, age, income, and other detail. See the example persona in Figure 3-1. The idea is to make the customer come alive in the minds of the Web team.

While the clients have a good idea of the characteristics, wants, and needs of their customers, for projects that can afford it, you can schedule workshops with actual customers to better get into their heads. These workshops make your personas that much more accurate.

Helen Johnson
Primary customer

Age:	23
Occupation:	Web Designer
Location:	Chicago, IL
Income:	$45,000

Background:
Helen is a 23 year old recent graduate of Northwestern University living outside Chicago. She is unmarried but has a boyfriend that she has been dating for 2 years who just got accepted to UCLA Law School and will be moving in a few months. She lives with her roommate Lisa and has 1 cat named Shamu, but she wants to follow Matt to Los Angeles. She interviewed at an entertainment company and just received an offer. She needs to start in 2 weeks. She desperately needs to figure out how to move the little bit of stuff she's accumulated since college, in one week's time, for an affordable price. She's a savvy online shopper and researcher, and is confident she can find everything she needs for her move online, including a new apartment.

Computer savvy:	high
Web savvy:	high
Online shop savvy:	high

Figure 3-1: A persona gives a name and a face to each type of customer.

Scenarios

Scenarios are plausible situations involving each persona. For example, a scenario for "Helen" might be as follows. Helen is moving to accept a new job in Los Angeles. She has accumulated enough stuff throughout her first couple years out of college that her car just won't do. She needs to find an affordable moving solution for her small amount of stuff that's not a semi.

From this example, you can see that an online moving company targeting this customer needs to quickly convey multiple affordable options from small vans she can rent to full-service, small-scale moving services.

By thinking through a couple different real-life scenarios, you have a better context from which to design the site's architecture, suggest rich media that serves a real purpose (and not just looks cool), and a content strategy that anticipates and meets the customer's needs.

Building an Outline for Your Site

Now, you must balance everything you learned about the target customers and their needs alongside the list of business requirements and start sketching an outline for the site. From this outline, you can build a *site map* (a flowchart-like diagram) that shows how you can organize all the content in the site.

Making a wish list of content, bells, and whistles

In no particular order, brainstorm a list of content and feature ideas:

- ✔ **Features** are the bells and whistles part of a Web site. For example, this may be a virtual tour of the company's facility, a search function, or a Contact Us form. Basically, a feature is an activity for the user.

- ✔ **Content** is the informational stuff, such as investor relations and product info.

Categorizing and prioritizing information

After you have a list of content and features, group them into an old-fashioned outline. (You remember, the ones that have Roman numerals in front of the main ideas.) This outline, like the sample shown in Figure 3-2, ultimately helps you build a site map.

The key to producing a successful outline is your ability to group similar items and prioritize ideas based on the user and business needs of the site. As you become more familiar with the list of content and features, a pattern begins to emerge. Some items go together quite easily, whereas other ideas don't fit in at all.

Here are the steps for converting your wish list into a workable outline:

1. **Group ideas.**

    Find features and content that seem similar and place them next to each other. For example, investor information and partner information fit nicely together. Set these aside and give them a temporary group title, such as About the Company. This group title can ultimately become a

navigation choice, and the stuff inside the group might all be accessed via a drop-down menu. Think of each entry in your outline as one Web page of content.

2. **Limit the depth of each group.**

For usability reasons, have no more than two levels of subcategories within each group. If you get deeper than two, you run into navigation challenges. Here is an example of two levels of depth within the About the Company group:

> 0. **About the Company**
>> 1. **Contact Us**
>> 1. **Partners**
>>> 2. **Clothing partners**
>>> 2. **Footwear partners**
>> 1. **Press**
>> 1. **Our History**
>> 1. **Executive Team**

3. **Group global content and features.**

If the list contains things like a member sign-in feature, a search function, or privacy policy info that should be accessible from anywhere on the site, set them aside in a Global group and don't worry about them not relating to one another just yet; you can categorize and group these later.

4. **Assign a priority to each group.**

Not every content group you create has the same business and user importance. Put three stars next to the most important groups, two stars next to the medium important groups, and one star next to the least important groups. For example, the Products group might be more important than the About the Company group and get three stars. For the global group, you can assign stars to each individual item.
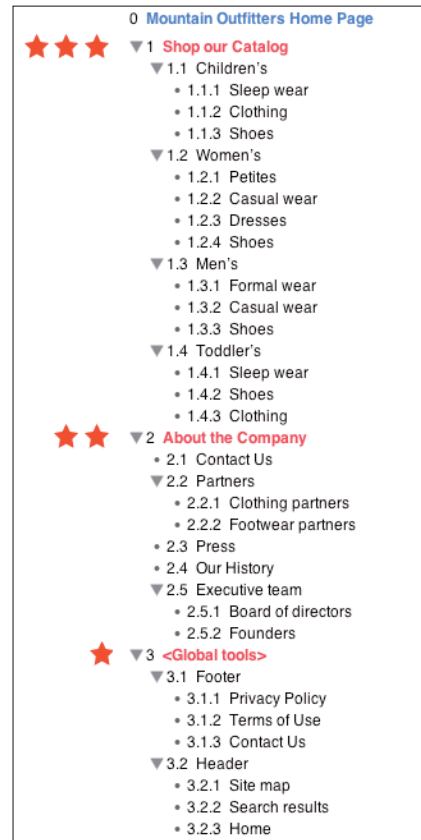
Figure 3-2: An outline of content ideas shows a number of content elements within each group along with stars to rank priority.

Following the five-to-seven rule

Assigning a priority with one, two, and three stars to each of your groups helps you determine which groups become your primary, secondary, and tertiary navigation. Primary navigation is the main set of choices that is front and center on every page of the Web site. Take a look at Figure 3-3. The four high priority groups (Children's, Women's, Men's, and Toddler's) from the outline in Figure 3-2 become the main navigation buttons on the site. You access the items within these groups via the drop-down menus. Secondary and tertiary navigation choices are also on each page of the Web site, but are visually smaller, off to the side, or found in the bottom footer. (You only need to use secondary and tertiary navigation sets as needed — they are helpful for organizing large content sites.)

Another tertiary navigation set

A secondary navigation set   A primary navigation set

A tertiary navigation set
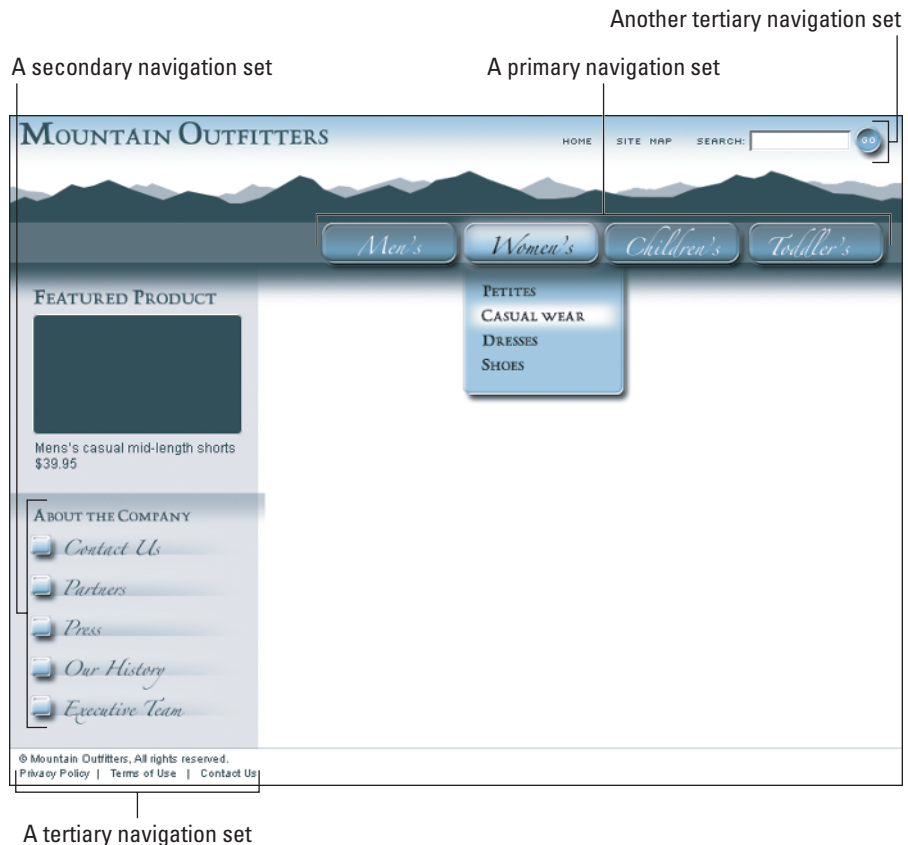
Figure 3-3: This page template shows you the design strategy for the primary, secondary, and tertiary navigation sets.

TIP

For usability reasons, be careful not to have more than seven groups within your primary, secondary, and tertiary collections. Studies have shown that users have difficulty getting their brains around a site when they are bombarded with more than seven items to choose from in each navigation set.

Also as illustrated in Figure 3-3, when you get to the Design phase, it's good practice to group all the primary navigation elements together in the page layout and give them the same visual treatment. This helps users identify each navigation set. Notice the primary navigation set at the top, the secondary on the left side, and the tertiary split between a few links in the upper-right corner and the footer links at the bottom.

Crafting a Site Map

WEB SPEAK

The outline you create for your Web site directly translates into the *site map*. The site map helps you visualize the structure of a Web site, called the *information architecture,* before you build it. To create a site map, you translate each main idea and subcategory from your outline into a diagram of boxes connected by lines and arrows to show how the pages interconnect. A site map is critical to the construction of your Web site, and you'll refer to it often throughout the development process.

If you're a print designer, think of a site map as a sort of thumbnail sketch that you'd make for a brochure design.

To transform your outline into a site map, follow these simple steps:

1. **Start with a huge piece of paper.**

   I like to start on paper because you can quickly sketch out ideas and you have plenty of available design space. Find a large piece of paper and work in landscape orientation. (You'll find that you need a lot of horizontal space.) After you work out the map's details on paper, you can re-create it on the computer to make a nice, clean copy that you can distribute to both the client and the team.

   When you're ready to transcribe your paper site map into a digital copy, try using a program like Inspiration (www.inspiration.com) or Microsoft Visio. This software allows you to quickly build visual flowchart diagrams. For you designers out there, you can also use Illustrator or Freehand (my personal choice). As shown in Figure 3-4, using one of these tools allows you to spruce up the site map with your own branding.

2. **Draw a box for each Web page.**

Starting with the home page, draw a box to represent each Web page of the site. Put the home page box at the top of the paper. Then, as shown in Figure 3-4, start a new row below and draw a box for each of your primary, secondary, and tertiary navigation group titles.

3. **Draw the subcategories.**

Begin a third row and draw a series of boxes for each page within the main sections. For space concerns, you may want to stack these pages vertically beneath their respective main idea boxes.

4. **Number the sections and subsections.**

An important step is to number each section so that you can refer to it more easily in the future and match it up with the official page index you create (as detailed later in this chapter). As shown in Figure 3-4, assign 1.0 to the home page and 2.0, 3.0, 4.0, and so on to all the navigation sections. Within each navigation section, label the subpages as 2.1, 2.2, 2.3, and so on. For your second level of navigation (pages under your subpages), use 2.1.1, 2.1.2, 2.1.3, and so on.
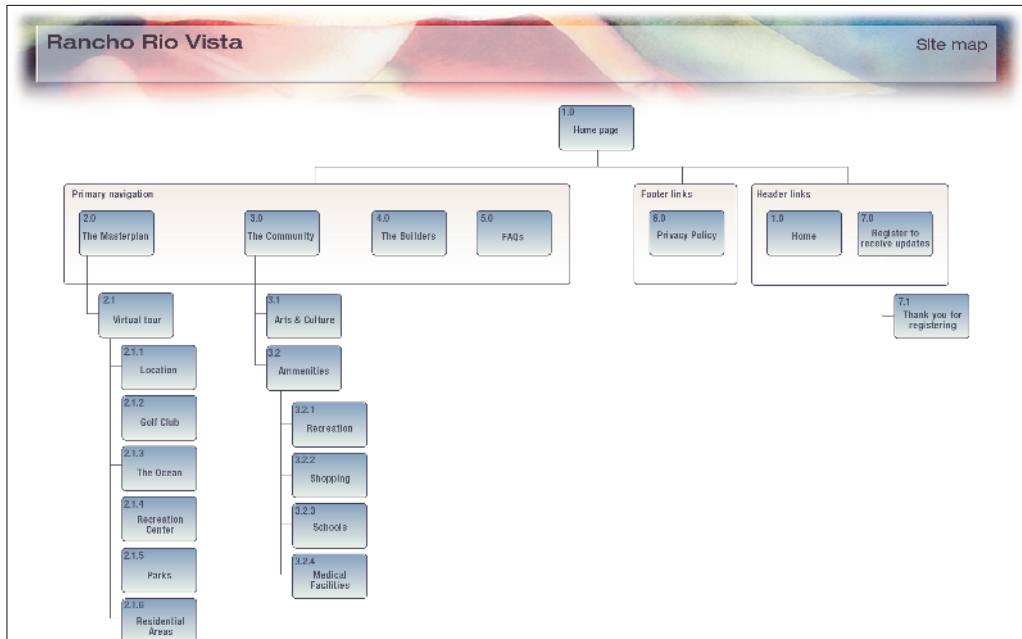
Figure 3-4: This site map, created in Freehand, has a unique header graphic that reflects the design company's brand.

WARNING!

A good site map foreshadows usability problems before your site goes into production. If your map has too many primary navigation categories that are each a little thin on content, you end up with a site that overwhelms the user with choices and clutters the screen. Conversely, if your site map has just a few primary categories that each have a ton of stuff, you end up with a site that takes forever to navigate through — making people click too many times to drill down to the info they need.

Remember, the best balance to have is five to seven choices within your primary, secondary, and tertiary navigation choices and no more than two levels deep of content within each category.

Reading between the lines and boxes

To show how the pages on your diagram link together, you must draw a series of lines. This is literally a game of connect the dots or, in your case, boxes. The following list includes some design conventions that many designers often use on site maps to show navigation, technical, and content plans for each page:

TIP

✔ **Direct links.** The most basic way to show how two pages link together is to draw a line that connects them. Generally, all lines sprout forth from the home page's box and connect to each of the main section boxes. Considering the amount of boxes that you probably have on your diagram, this can get messy quickly. To keep the map clean, draw one line coming from the home page and then have it branch into individual lines that connect to each subpage, as shown in Figure 3-5.

```
                    ┌─────────────┐
                    │ 1.0         │
                    │ Home page   │
                    └─────────────┘
                           │
          ┌────────────────┼────────────────┐
   ┌─────────────┐  ┌─────────────┐  ┌─────────────┐
   │ 2.0         │  │ 3.0         │  │ 4.0         │
   │ Products    │  │ Services    │  │ About Us    │
   └─────────────┘  └─────────────┘  └─────────────┘
```
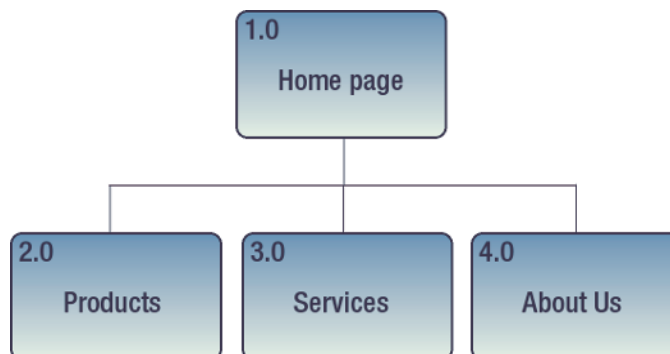
Figure 3-5: Clean up the connecting lines by using a branching approach.

✔ **Linking to a shared page.** In many cases, you have a few shared pages, such as a Terms of Use page, that many or even all the pages in the site link to. (In the Web industry, shared pages are called *global* pages.) In cases like this, drawing connecting lines from each page leading to the shared pages doesn't make sense and creates a big mess. Just as Alaska and Hawaii are traditionally shown pulled out on a U.S. map, you can create a special area on your site map just for the global pages. See how the footer pages are grouped together in Figure 3-6.
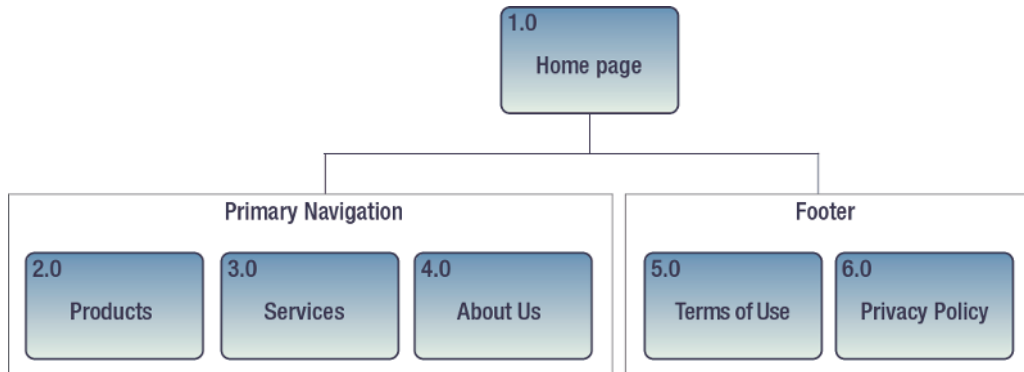
Figure 3-6: The site map groups global pages, such as Terms of Use and Privacy Policy, within a footer box.

✔ **Restricted pages.** If you have a password-protected page or section, draw a smaller box representing the login *gate* that precedes the area.

✔ **Automatic flow from one page to the next.** Sometimes you want a Web page to display for a few seconds and then automatically flow to another page without requiring the user to click. For example, many Web sites have an animated intro sequence that plays before you get to the home page (which I personally do not recommend because most people don't have the patience to sit through it). To accommodate situations like this on the site map, I use a squiggly line symbol. You can show automatic flow by drawing the beginning page directly on top of the page that it leads to. Then, draw a squiggly line with an arrow between the two pages, as shown in Figure 3-7. Figure 3-8 shows an example of a Web page with an animated sequence.
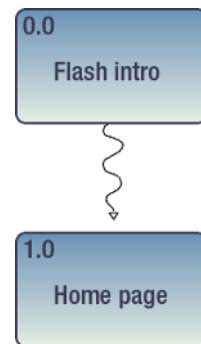
Figure 3-7: To show automatic page flow, draw the pages stacked, connected with a squiggly line.
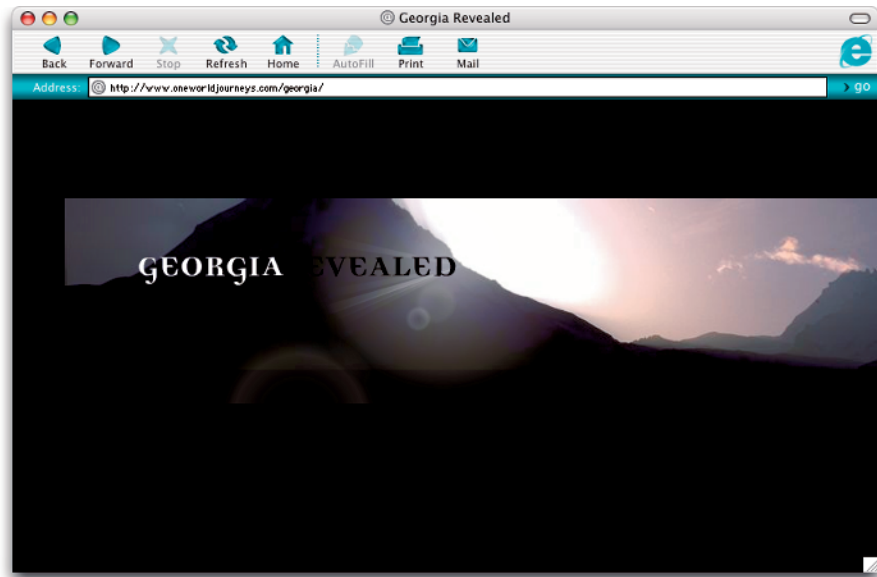
© OneWorldJourneys.com

Figure 3-8: This animated intro sequence plays before you get to the home page of this site.

Developing your own set of symbols

As you blossom into a seasoned Web designer, you develop your own lexicon of symbols, lines, and shapes to represent various elements and interactions of a Web site on your site map. When creating your own symbols, the only rule is consistency. Regardless of the symbols or line art you decide to use on your site maps, just be sure to use them consistently and to clarify their meaning in a legend.

Here are a few miscellaneous elements that you may encounter and some ideas for representing them on your site map:

- ✔ **Databases.** To represent a database on a site map, most design firms use a cylindrical container symbol that looks like a can of soup. The database "soup can" can go anywhere on your site map, but most often I see it placed in the center or on the bottom — wherever it's most convenient to link multiple pages to. Pages that access the database have a dotted line connecting to it, as shown in Figure 3-9.

- ✔ **Flash movies.** Flash is a software application that creates highly interactive animated movies and game-like applications that you can place on a Web page. Because Flash movies can be fairly complex and require their own concentrated development, you should have a dedicated symbol to represent them on the pages where they occur on your site map (people usually use the little "f" symbol that Macromedia uses to denote Flash movies).
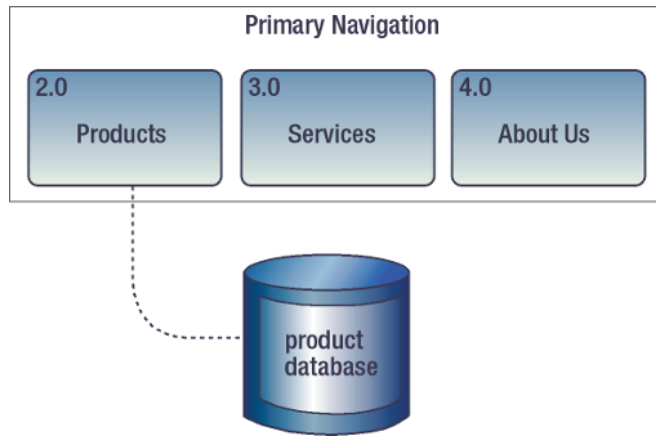
Figure 3-9: To connect a page to the database, draw a dotted line to it.

WEB SPEAK

✔ **Template pages.** Sometimes designing a master template page is the most efficient way of building a large number of Web pages. An online database *populates* or fills the template with data, depending on the user's actions. For example, an online store may have hundreds of products that all use the same template page for display. To indicate a template page on your site map, draw it as a stack of pages — like the one shown in Figure 3-10 — and remember to draw a dotted line to the database "soup can."

Everyone's singing from the same songsheet

Establishing clear communication between you, the team, and your clients is a high priority throughout the Web design process. Sharing the site map with the clients is an excellent way to make sure that you're on the right track and that the clients knows exactly what they're buying.

Many Web design firms have clients sign off on the accepted site map in order to confirm and manage their expectations for the site. This way, the client can't come back to you in the future asking about a "missing Founder's Bio page." Armed with the client-approved site map, you can simply remind the client that the page in question was never part of the deal.

Although this sounds harsh, remember that the sign-off works both ways! If the site map included a Founder's Bio page, the client has every right to wonder where the page is in the developing design. In addition, an agreed-upon site map helps the design and HTML production teams anticipate and plan for all the pages of the site so nothing gets left out. Make sure that each

production team member working on the site has a copy of the site map that they can refer to throughout the project. Ultimately, a site map is a great tool to keep both sides singing from the same songsheet, so to speak.
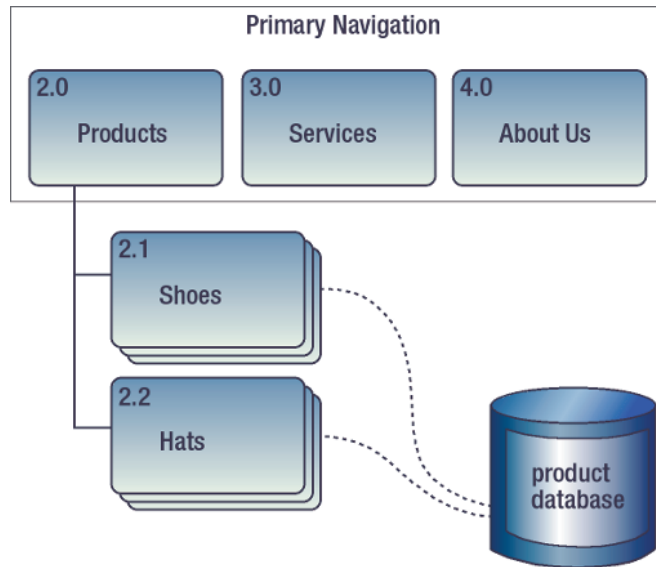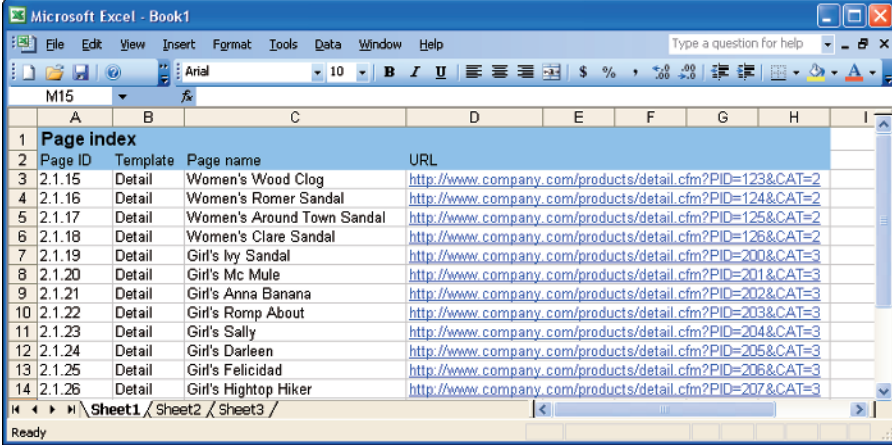
Figure 3-10: Indicate template pages on your site map by drawing a stack of pages.

The official page index and naming conventions

Another function of the site map is to establish an official list of each page, a.k.a. the *page index,* that contains all the naming conventions. I like to make a spreadsheet that has a few columns of information such as the page's common name (About Us), its technical name (aboutus.htm), and even its URL path (http://www.company.com/products/detail.cfm). That way, the development team can know the directory structure of the entire site before building it.

TIP

For large database-driven sites, the page index is extremely important because you can't possibly visually show all possible pages in the site map. As you can see in Figure 3-10, the site map shows the individual product pages as a generalized stack-o-pages. The page index, however, in Figure 3-11 lists all 100+ products in a spreadsheet format. Also notice how the page index uses the numbering scheme from the site map so you can match up pages between the two documents.

Figure 3-11: The page index shows all pages planned for the site. Notice how it references the numbering scheme of the site map.

Building a Map for a Site Redesign

Many people assume that Web design is always about creating a new Web site from scratch. The truth is that almost every organization of any size already has a Web site. The primary need, therefore, is to keep the existing Web site fresh and evolving with the latest technologies and the changing needs of the company. Therefore, more often than not, clients approach you to *redesign* their Web site.

TIP

Site maps come in handy not only in the design process, but also in the redesign process. They allow you to see the current site in its totality and how the pages relate to each other. With a site map, you can see where the problems lie, and you can get ideas for a better design.

Deconstructing a Web site

So how do you create a site map for an existing Web site? You build it by picking the site apart to see how it's made. Here are four easy steps to get you started:

1. **Start with the home page.**

   Go to the home page of the Web site and look closely at all the links on the page. On a large piece of paper, draw a box at the top for the home page and, to the side, jot down a list of all the links you see. Keep them

grouped as they are on the home page. For example, under a Footer title, list all the footer links. This list helps you stay oriented in the site as you pick it apart.

2. **Identify the main categories.**

Click all the home page links and see where they take you. Your goal is to identify the purpose of each link and to distinguish the links that seem to be the most important main categories. You may be surprised to discover that some important content is hidden in a subtle link not grouped with the main links. This may be why you are redesigning the site!

For each of the main categories you discover, draw boxes in a row below the home page. In a third row, draw boxes for all the content within the categories.

3. **Represent the global functions.**

Some links are consistent on each page. For example, each page may have a link to a Privacy Policy or a Contact Us page. Identify all the shared global pages and draw them in a separate area on the site map.

4. **Look for database interactions.**

Keep an eye out for pages that rely on a database. These are easy to spot — look for pages that collect data. The data is obviously going somewhere when the user submits it. Also look for pages that have a funky URL. If the URL has a long string of weird characters like $ and ?, it's a sure sign of a template page filling itself with content from a database.

Also, ask the client how many databases the Web site uses and for what purpose. The client may have a CMS (content management system) for all images and text, have a database for products and prices, and a separate database for customers. Most likely, you need to incorporate these existing databases in your redesign. If not, you need to know how your client plans on upgrading these systems.

Finishing the site map

Completing the site map for a redesign involves the same process as creating a site map for a brand-new site — use your set of symbols and line art to fill in the detail of what's happening in the site. In a redesign process, you don't have to get too detailed in your deconstructed site map — save your energy for the redesigned site map. The purpose of making a site map for the old site is to give you a better idea of the site's current information architecture. It serves as a good point of discussion with the client so you can ask a lot of questions and formulate ideas for the redesign.

Here's a sample list of questions to ask the client:

✔ **What do you like least about the site now?** Find the root of the current site's problem by probing to see if the information architecture of the site or the content needs revising. Maybe both are fine, and the client just wants a new look to the site or to add new features such as interactive Flash elements.

✔ **Should any new content be added, or old content removed?** If you're adding or removing a great deal of content, the site's information architecture may need a substantial redesign.

✔ **Has the site's purpose changed?** Many times a company has an existing Web site that was originally designed to act as a marketing piece, but now must act as a revenue-producing business machine. If so, you are faced with a substantial redesign of both content and architecture.

✔ **Has the company changed its focus or market positioning?** In such a case, the site's content and navigation, as well as its look and feel, may change drastically. When you design sites like these, you are almost starting over from scratch.

Developing a Marketing Plan

After you thoroughly define your target audience for the Web site, it's a good idea to get started on the marketing plan for the site. This planning can begin to happen in parallel to the site's development.

Do not underestimate the importance of marketing Web sites. The Internet simply has too many Web sites for anyone to find — or care about — your site unless you bring it to their attention.

Therefore, you or a dedicated marketing person on your team must sit down with the client and brainstorm a plan. Although some clients may not know the best way to market their Web site, they do know a lot about reaching their customers. Together your team and the client should develop a list of ideas, action items, and a budget.

Offline marketing

The most effective way to market a Web site is to combine *online* and *offline* marketing campaigns. *Offline marketing* refers to all media that's not on the Web — radio, magazines, event sponsorships, trade shows, and so on.

Here are some offline marketing ideas to consider:

- ✔ **Magazine articles.** Editorial articles in professional trade magazines are highly effective marketing tools for a Web site.

  If your marketing plan includes getting exposure in magazine articles, remember that most magazines plan their editorial schedule well in advance. Enlist the help of a public relations specialist who is familiar with magazine schedules and who has the contacts to get your client's Web site into some headlines.

- ✔ **Radio.** Radio is a good way to focus on regional areas. You just need to keep in mind your target audience. Local news programs that feature the weather and traffic reports are great ways to capture the attention of the nine-to-five worker bees during commuter hours. Advertising on "soft hits of the '80s and '90s" stations is good for targeting office workers, whereas the local hipster music station can attract the young and young at heart.

- ✔ **Trade shows and conferences.** Trade shows can be an expensive way to promote your Web site. Not only do you have to consider the price of the booth space, but also the cost of a nice display, travel, and hotel. Nevertheless, you may find that they work perfectly for your site's needs. In addition to trade shows, you should look into conferences. If possible, try to get a speaking gig at the show or enlist one of the speakers to talk about your site.

Online marketing

Online marketing refers to a Web-based campaign (for example, buying those annoying banners at the top of Web pages). Here are some ideas for an online marketing campaign:

- ✔ **Online feature stories.** Every industry has its share of informational or resource Web sites that run feature articles. Snuggling up close to one of these sites is a great way to promote your site.

- ✔ **Ad banners.** Most advertising folks claim that online banner ads have a 2 percent *click-through* rate, meaning that 2 percent of people who see the ads actually click them to visit the respective site. So if you're considering this option, be honest with yourself: When was the last time you clicked one of these ads in your last 100 Web visits? Clever ads, however, can at least create awareness in the marketplace.

- ✔ **Search engines.** Most people find new Web sites by using one of the search engine sites such as Yahoo! or Google — my personal favorite. When you type a term in the Search field of one of these sites, up comes a listing of sites that match that keyword. The search engines look in the

<meta> tags, page titles, and actual content of each HTML page for such keywords. You can also purchase keywords at search engines to help ensure that your site appears in the resulting list when people search.

✔ **Link exchanges.** A cheap way to market your site is to swap links with a partner Web site that shares a similar audience. Exchanging links is a common practice: You link to the partner's site either with a mini button, a text link, or a banner, and your partner reciprocates with something similar linking to your site.

TIP

If you're going to run a series of ad banners, they are most effective when run consistently for at least three months and when you run them on sites that share your target audience. By some estimates, a user must see the same ad nine or so times before it sticks and sways him or her to pay attention.

Organizing and Navigating Web Content

*T*he Web is a unique communication tool — it's not like the telephone, TV, books, or marketing brochures. People interact with Web sites differently than with other mediums. They click buttons and links, access drop-down menus, fill out forms, and interact with animation, audio, and video media. For these reasons, to design Web user interfaces that simulate other media is to ignore the advantages of Web technology. Designing a Web site to look and function like a book, for example, is a waste of interactive media's capabilities.

As a budding Web designer, your biggest challenge is to think through the interaction design of each page in your site in a way that maximizes the Web's abilities, presents content on the page in a logical and consistent manner, and allows users to quickly get from one page to the next. In this chapter, you continue in the role of Information Architect and dive from the site map view down into the page-level view. You find out how to create a wireframe diagram for each page that shows its content and interaction plan, and I discuss ideas for navigating from one page to the next. At this stage of the game, you're effectively creating the blueprints that show how to build the Web site.
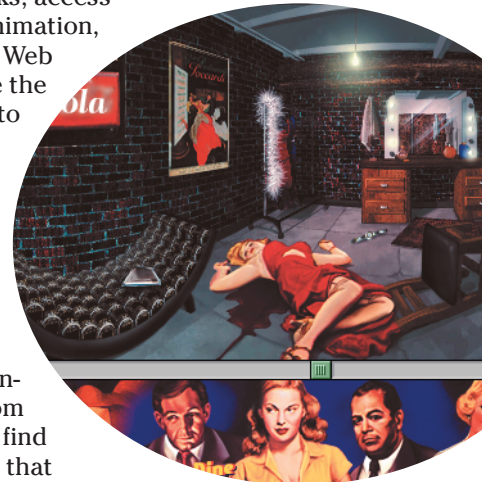
Page-Level Planning

While the site map shows you the 35,000-foot view of your site and how all the sections work together, it doesn't show you the page-level detail. What goes on each page? How much content? How can people navigate within the page and to other parts of the Web site?

To answer questions like these, you create a wireframe for each page of your site (or each unique page layout). A *wireframe,* such as the example shown in Figure 4-1, is a diagram view of a page. As a designer, I like to use a program such as Illustrator or Freehand to build wireframes, but information architects generally prefer Microsoft Visio as they do for developing site maps.
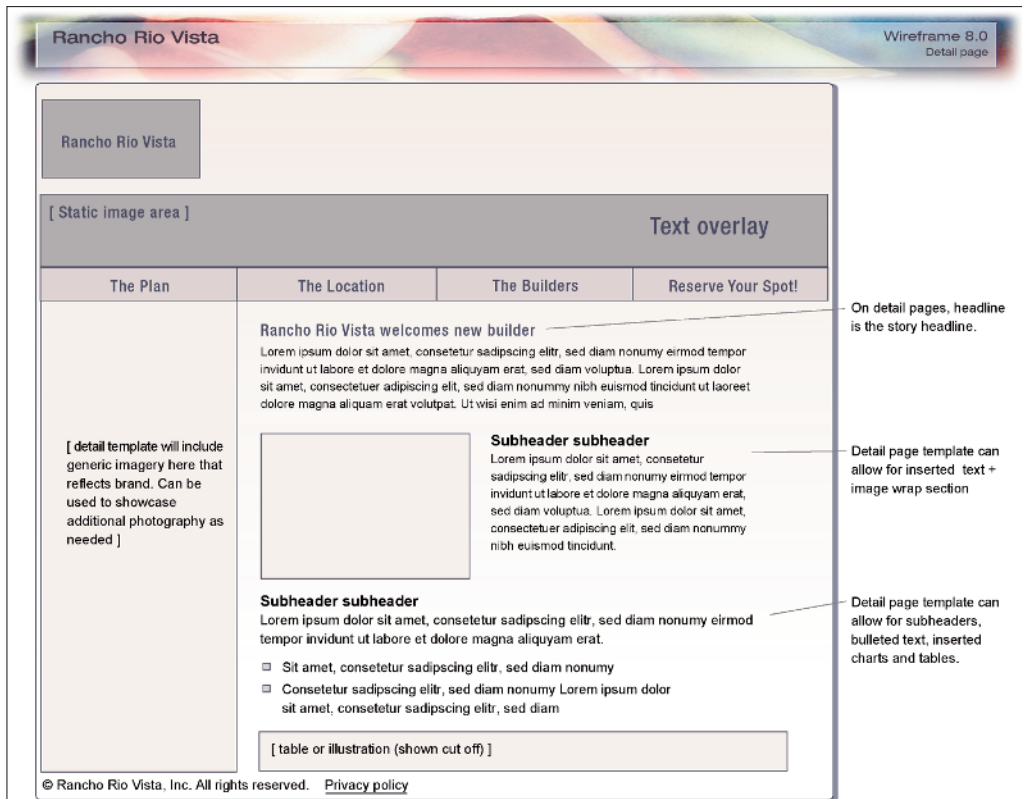
Figure 4-1: A wireframe diagram shows the content and interaction design structure of a Web page.

Mapping out content zones

Working out what goes on each page and how the content is presented and navigated is a huge task, especially for large enterprise sites. So, at one company I worked with, I instituted a pre-wireframe stage and created a template, shown in Figure 4-2, to help the team quickly map out content zones for each page to make sure they all worked well together and were consistent.

The template included key information to help remind the team of the page-level goals. For example, a key user goal for the Products page may be the ability to quickly find a certain product. A key business goal may be to showcase a new product. With knowledge of these goals, you can define a few content zones and show their relative placement, sizing, and priority on the page. Figure 4-2 shows the template filled in with content zones and a few key content elements. Notice how low priority elements fall *below the fold* (out of initial viewing range on the Web page, requiring users to scroll to see them).
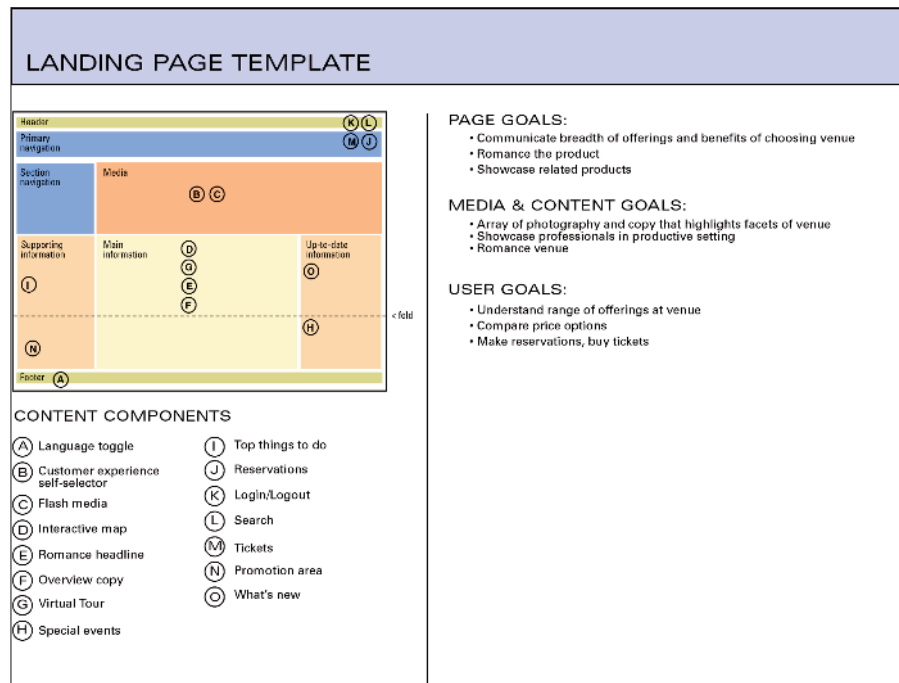
Figure 4-2: A pre-wireframing template is a great tool to help a team quickly map out appropriate content zones while keeping business and user goals in mind.

Wireframing

After a pre-wireframing exercise, you can fill out the detail within each content zone you identified. (A pre-wireframing round is an optional step recommended for large, complex sites. Most sites just go right to wireframes from the site map.)

Wireframes serve sort of like your laundry list of what goes on each page from a content and media perspective. Here are some examples of what your wireframe might show:

- ✔ What copy goes on the page and how long it is
- ✔ Which CMS objects go on the page (if you're using a content management system)
- ✔ What kind of images and media (such as Flash and video files) go on the page
- ✔ Your technical and interaction plan for the page — links, widgets such as drop-down menus, check boxes, and Submit buttons, and DHTML (Dynamic HTML) implementations like CSS (Cascading Style Sheets)

I create wireframes at-size, meaning they are the same widths and heights of my final Web page. (As of this writing, 800 x 600 is the lowest common denominator size for a Web page — which accounts for laptop viewing.) That way, I can get a real sense for how much content comfortably fits on each page before a user has to scroll. Otherwise, you may have trouble at the Design stage trying to squeeze all the stuff required by the wireframe onto the page. Or conversely, the page can look sparse if in reality the content doesn't take up as much room as the wireframe suggested.

Wireframing is an important step that you should not leave out of any interactive project — whether it's for a mobile project, a DVD, a Flash presentation at a tradeshow, or of course a Web site. Like a good architectural blueprint, the wireframe is your opportunity to work out all the content and interaction design issues on paper with not only the client, but also the technical team who builds the site.

You don't need to build a wireframe for every page of the site. You need only to make a wireframe for each unique layout. For example, if you have 100 product detail pages in your page index, you can build just one wireframe that shows how each page works. If one or two product pages have a slightly different layout or content twist to them, however, it's a good idea to make a special product detail wireframe just for them.

Presenting Content on the Page

When building your wireframes, remember that you are not limited to a static presentation of text and graphics. Many technologies — from Flash animation to DHTML — help you maximize your page space and present content in more interesting and interactive ways. In addition, your wireframes should also account for displaying copy on the page whether it's embedded in the HTML page or coming from a content management system. In this section, I discuss insights for presenting different kinds of content in your wireframes.

Indicating text on a wireframe

Because you're sharing wireframes with clients and team members alike, I indicate all navigational elements and headlines for major content sections with readable text to orient them. All other copy components can be indicated with *greek* (unreadable) text that reflects their allocated character count (the number of letters, punctuation, and spaces) so you get a sense of actual text block size. Figure 4-3 shows an example of integrated readable and greek text.
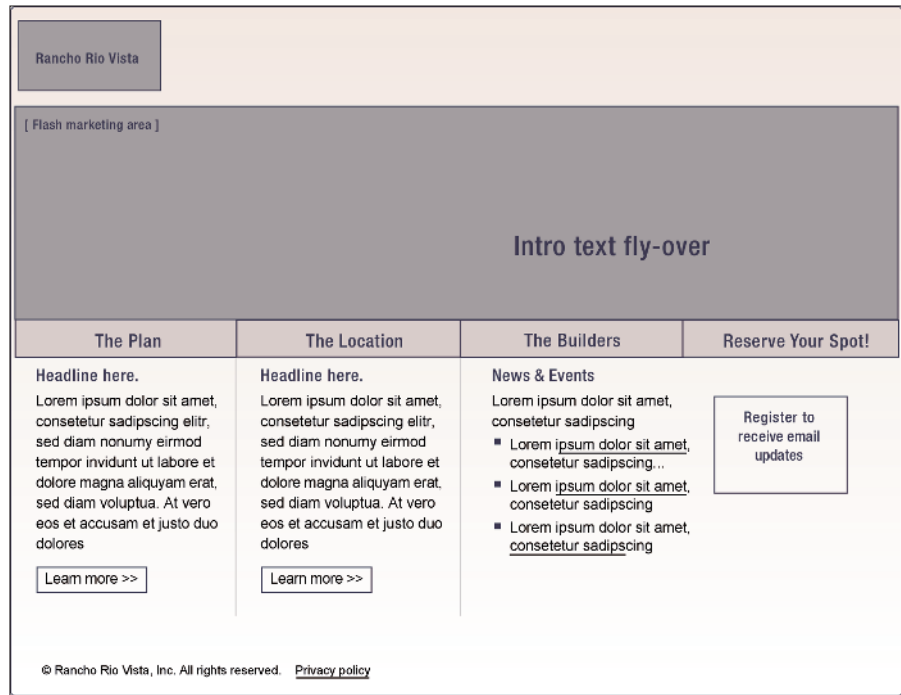
Figure 4-3: Use legible text for navigation and headlines to orient people and greek (unreadable) text for everything else on your wireframe.

You can search the Web for **lorem ipsum generator** to find sites that generate paragraphs of greek text — and even specific character counts — that you can copy and paste right into your wireframes.

WARNING!

Be clear with clients and team members alike that the readable text is for communication purposes only and that final copy comes from the writers in due time. I've found placeholder text on high-profile Web sites upon launch because the developers simply copied what they saw on the wireframes and no one caught it!

Working with a content management system

A content management system is a software and database combination. The software provides an administrative front-end to manage content stored in the database. The benefit of using a CMS to store text and graphics is that once you define a bunch of content types like "short description," "product name," and "product thumbnail image," you create instances of these objects, fill them with actual copy or images, and then assign them to one or more Web pages. That way, if you update the instance in the CMS, the content automatically updates wherever it's used on the site. This strategy separates the content from the Web page presentation to give you more flexibility for future design and content changes, making the site easier to maintain.

If your site uses a content management system, a Content Strategist works closely with the Information Architect (on smaller teams, this can be one and the same person) to first identify all the different content types a site has and then determine which content types go on each page. To keep track of all the content types, their instances, and which pages they reside on, you should create a content inventory spreadsheet for larger projects.

Maximizing your space

REMEMBER

Most people have small computer monitors. If you hold up an 8½ x 11-inch piece of paper horizontally, you're looking at the size of most people's window to the World Wide Web. After you factor in the browser's interface with all its buttons across the top, favorites tabs, and scroll controls, the viewing space devoted to the Web is whittled down to a postcard size, or about 800 x 600 for most users. This scrawny window is your Web design canvas.

Here are some organizational and interactive strategies to help you better present your content on this wee little canvas:

✔ **Scroll to see less important content.** You simply can't shove everything into the viewable area — the first 800 x 600 pixels of the browser window. So, place the high priority items within that initial window, and lower priority items below the fold (described in the earlier section "Mapping out content zones"). During the zoning exercise in the pre-wireframing stage, you decide which elements are above and below the fold.

TIP

Give visual clues to let users know more stuff is below the fold by purposefully showing headlines or the tops of images that lead to more stuff below. Users don't mind scrolling, just as long as the page doesn't go on forever. A good rule is to limit the total page height to twice the initial viewable height (so 1200 pixels for a 800 wide x 600 tall page).

✔ **Scroll horizontally too.** You can use technologies like Flash to present horizontally scrolling content. This technique is not often used, so it can be a fresh solution to presenting a lot of content. Take a look at Figure 4-4. The Flash interface has enough visual clues — also called *affordances* in highfalutin circles — to let users know they can explore side to side and even up and down for a panning type of experience.

Figure 4-4: On this page, a horizontally scrolling panoramic interface maximizes the screen space.

TIP

✔ **Layer your content.** Another way to maximize the viewing space is to reveal layers of content on top of the Web page as the user rolls over or clicks something with the mouse. You can hide and show content layers through technologies like JavaScript, DHTML (Dynamic HTML), and Flash. Typical uses of this strategy are expanded context views (shown in Figure 4-5) and drop-down menus like the one shown in Figure 4-6.

✔ **Design a revolving door of content.** Because Web page space is at a premium, another dimension that you need to consider is time. If you can't fit everything that you want inside the viewable area, you may want to think about animating the area. This way, you can cycle a lot of clickable links in and out over time in a relatively small space rather than crowding them all on the page at once. Animation also works well with navigation choices. Take a look at Figure 4-7. In this example, a number of different promotions are showcased in one small area.

Figure 4-5: A content module expands to reveal more information when the user clicks it. Otherwise, it is neatly closed and does not clutter the page.
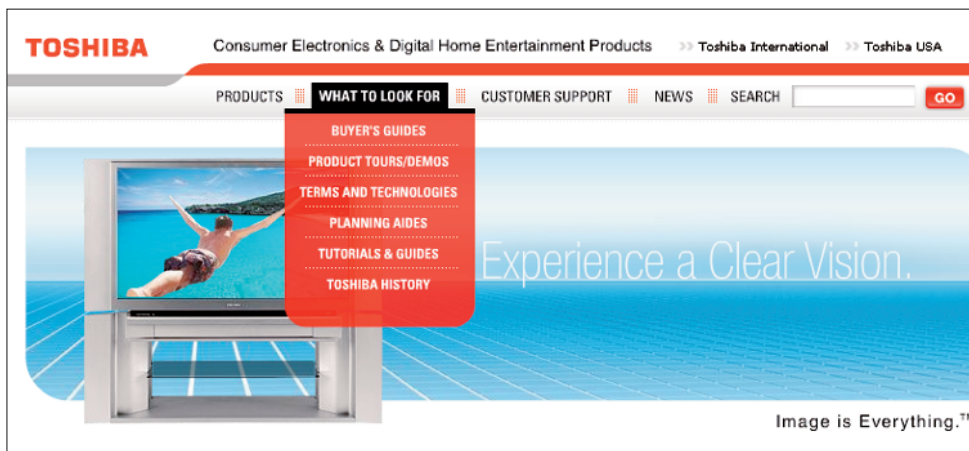
© Toshiba America Consumer Products

Figure 4-6: DHTML can produce a drop-down menu to appear when a user rolls over a navigational item.
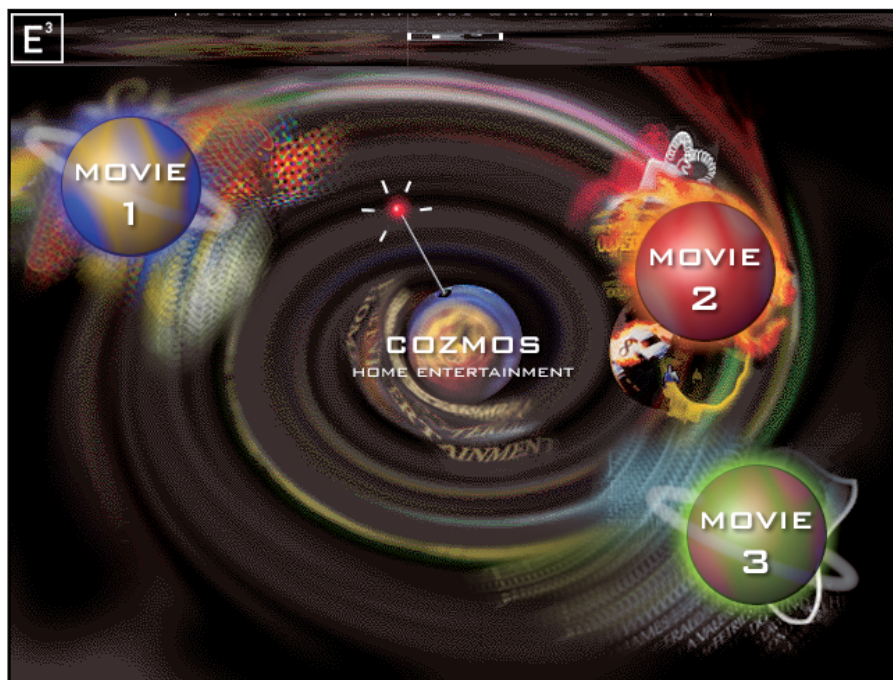
Figure 4-7: This animated promotional ad can provide a rotating presentation of links to new products.

Indicating Flash in your wireframes

Sometimes Web pages contain highly interactive, animated components built with technologies like Flash. These interfaces can have a lot of information and choices embedded within them. For example, you can build an entire application with Flash — including its own collection of screens, links, and functions — and place it on just one Web page. Take a look at a beautiful Web site about the nation of Georgia (www.oneworldjourneys.com/georgia/). A large portion of the experience happens in just one Web page with a Flash movie, as shown in Figure 4-8. Though the interface looks like a typical Web page, it is actually one of many screens contained in one Flash movie, playing on one Web page.

By using Flash, you can optimize the page's performance because Flash is highly efficient at compressing and displaying graphics. The slight downside, however, is that users need the Flash plug-in installed in their browsers to view it (although I think the latest figure is that 99 percent of users have the Flash plug-in).

YEW TREES ARE THE DIRECT DESCENDANTS OF THE VERY FIRST TREES THAT APPEARED ON THIS PLANET 150 MILLION YEARS AGO.... ONLY IN BATSARIS DOES A WHOLE FOREST SURVIVE, WHERE THE TREES ARE BETWEEN 1500 AND 2000 YEARS OLD.

Continue ⟫

[expedition map]   [today's notebook]

© OneWorldJourneys.com

Figure 4-8: With Flash, you can create an entire interactive experience on just one Web page.

Because Flash has so much interaction going on, your Web page wireframe can't possibly capture it all. Therefore, simply indicate a Flash element by drawing a box representing its relative size and shape. Make a note on the wireframe that the box is a Flash element, indicate its name, and reference any associated site map and wireframe storyboards that it may have just for itself.

Yes, that's right, because Flash movies can be so complex, they can have their own site map and *wireframe storyboards*. Wireframe storyboards are just like Web page wireframes except that you have a series of wireframes like a story-board that shows a progression of interaction in between states or sections of the Flash movie. As shown in Figure 4-9, Flash wireframes can also have visualization diagrams on them to show dynamic interactions such as a 360° panorama with embedded interactive points.

TITLE FOR SELECTED ROOM 2   EXPLORE HOTEL: SELECT... ▽

Each room is a 360°
panorama displayed
within this media space.

Hotspots within the
panorama link to
another panorama.

A different icon reveals
additional photography
upon rollover.

ROOM 1   ROOM 2   ROOM 3   ROOM 4   ROOM 5   ROOM 6
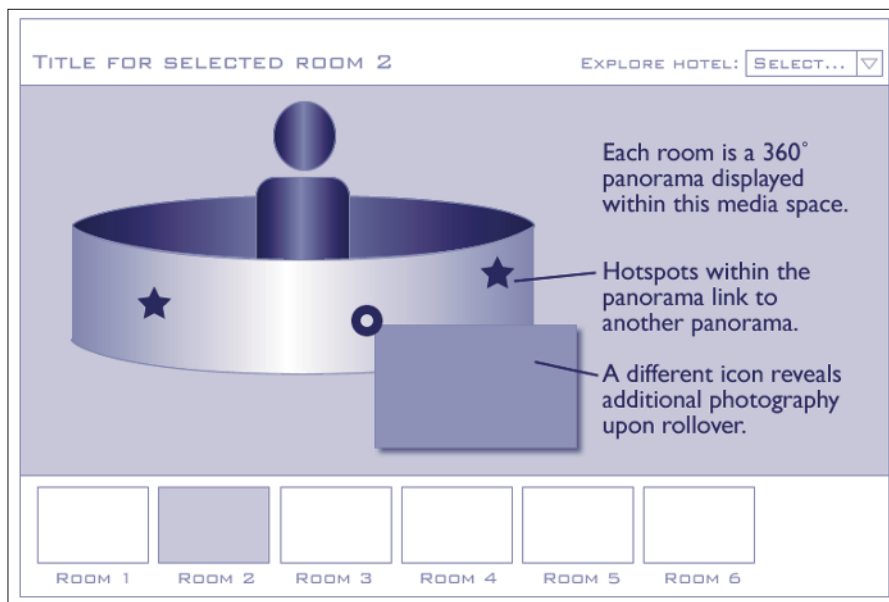
Figure 4-9: Flash wireframe storyboards show dynamic interaction models such as this interactive 360° panoramic space.

Creating flexible interfaces with animation

One entertainment client wanted its Web site to showcase a variety of movies on the home page. Each month this client wanted to show a different collection of movies — some months, only four movies; during other months, as many as ten. This simple requirement, however, would result in an interface that looked sparse one month and overcrowded the next, which presented a big challenge for my design team.

The solution was a modular, animated Flash media space on the home page that rotated through all the choices. The interface showed no more than three to four choices on the screen at any one time. In addition, the modular approach made updating easy. Each month, the client simply built icons for the new movies and put them into rotation. The client liked the approach and decided to create a similar animated interface to showcase its interactive games at a trade show.

Getting Around in Style

Other than the visual design, determining how a user will navigate the site is one of the more challenging creative tasks that you face. The main goal is to make people feel in control of the site and capable of getting around quickly and efficiently. Nothing is worse than a user feeling lost in your site.

To help people get around and stay oriented, your navigation scheme must be a road map of the entire site, complete with "you are here" signs. In this section, I discuss a number of tips and standard design conventions to keep in mind when designing your interface.

For the jet set: Global navigation

Remember the famous line from the '70s TV commercial: "How many licks does it take to get to the Tootsie Roll center of a Tootsie Pop?" With candy, the more licks, the better. On the Web, however, the opposite is true. Users won't find anything tasty about navigating through gobs of pages. Your goal is to get users from one page to the next in as few clicks as possible.

TIP The best way to reduce the number of clicks is to provide the same set of navigation links on each page of the site. This strategy, called *global navigation,* enables people to quickly navigate from one main section to the next without needing to retrace any steps by clicking the dreaded browser Back button.

A global navigation system also gives people a sense of the Web site's size. For example, Figure 4-10 shows that you're in the Breads section and that you can explore three other main sections (Meats, Dairy, and Fruits & Veggies). Global navigation also provides a mental anchor. Putting the same global navigation scheme in the same place on every page creates a point of reference that helps people stay oriented in your site.

Section navigation

After users select a category from the primary, secondary, or tertiary navigation group (see Chapter 3 for more on navigation groups), they're transported into a section. Assuming that the section has a few levels of content within it, you need a way to navigate through it. A typical practice is to reveal a set of *section navigation* choices on the page. These choices are unique to their section, but the region you select to display them is the same region used to display section navigation for other areas of the site.

Figure 4-10: With a global navigation system in place, users can quickly traverse from one main section to the next.

Figures 4-11 through 4-13 illustrate a few common ways you can represent section navigation. In Figure 4-11, the user clicked the Products link and sees the first level of section navigation displayed on the left. As shown in Figure 4-12, if the user clicks the Shoes link, the user goes to the Shoes page, and the section navigation expands to reveal a second level of navigation within Shoes.
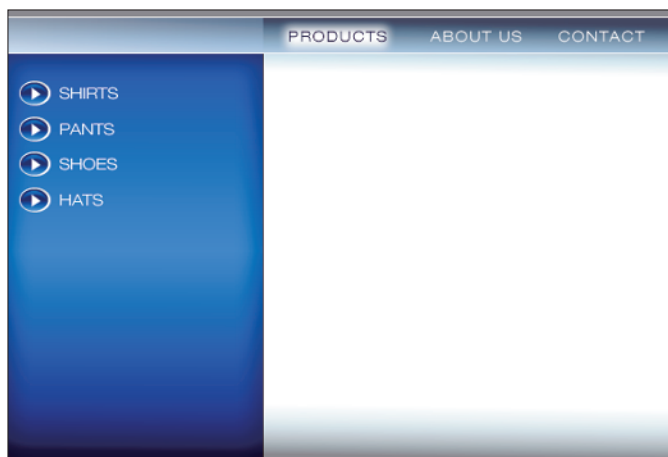
Figure 4-11: This section navigation design reveals the first level of pages within the Products area of the Web site.
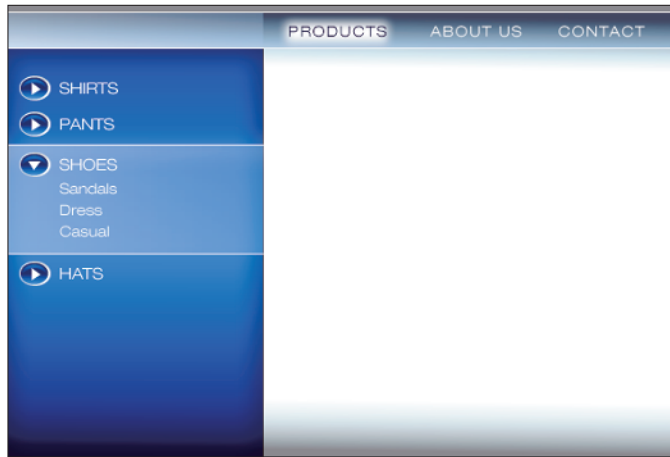
Figure 4-12: This section navigation design reveals both the first and second level pages within the Products area.

Alternatively, as shown in Figure 4-13, you can display the section navigation fully opened. The advantage is that users can quickly see all the first and second level content with a section. The disadvantage is that the navigation can take up a lot of room and look cluttered and overwhelming to the user.
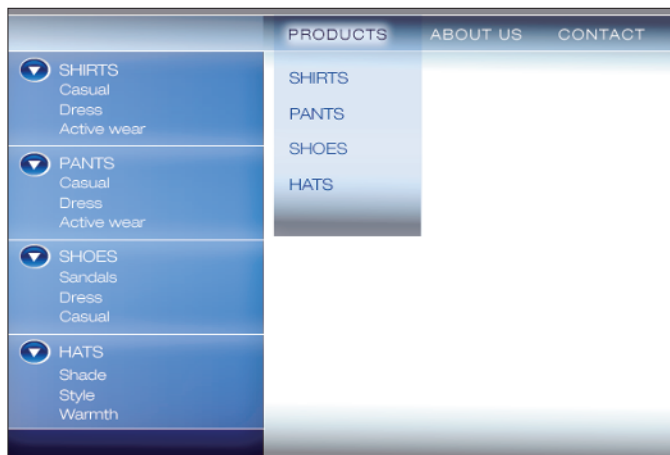
Figure 4-13: Section navigation is often shown as a drop-down menu as well as on the page itself.

Figure 4-13 also shows how section navigation is often included as a drop-down menu that displays when the user rolls over a main navigation choice. Typically, just the first level of pages is shown, but *flyout* menus can provide quick access to second level pages. It is okay to have redundant section navigation — showing it both on the page and including it as a drop-down menu in the global navigation system.

For nature lovers: Leaving a trail of bread crumbs

If Hansel and Gretel can use a trail of bread crumbs to find their way back through the forest, just think of what a digital version of bread crumbs can do for visitors to your Web site. *Bread crumbs,* as they're called in the Web design industry, are literally a trail of text links that record your steps as you go deep into a section, as shown in Figure 4-14.
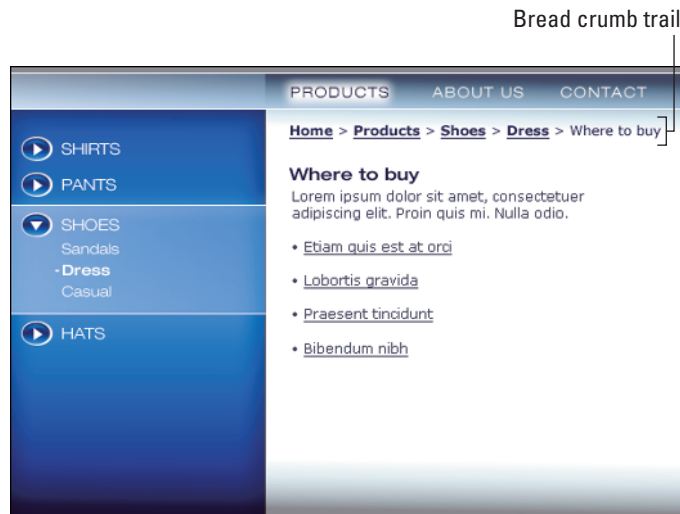
Figure 4-14: This bread crumb trail provides a convenient way for you to retrace your steps in a site.

Bread crumbs are helpful to navigate sites that have more than two levels deep of navigation within a section. For example, if you use the section navigation to go down to a page that has even more links on it, you now are diving into a third or fourth level of the section's hierarchy. It's just not practical to display third and fourth level navigation on the page. So, the bread crumb can simply record your steps and get longer and longer as you dive deeper (hopefully no more than four levels or you are seriously getting into the catacombs of a site!).

Each bread crumb link provides a quick way to retrace your steps back up the hierarchy. You don't have to follow the links in sequence; you can click any link in the trail to quickly jump back to a different level in the hierarchy. The trail also gives you a good idea of where you are in the site. Unlike with a global navigation scheme, however, you cannot jump across to *another* section; you can only jump back to a level *within* a section. Think of a bread crumb system as the browser's Back button on steroids.

TIP

Graphically, the design convention for a bread crumb interface is to, in a simple row, show each previous step as an underlined, active text link followed by an arrow, colon, or pipe. The last link at the end of the trail is *not* underlined and represents the page that you're currently on. This last link is not clickable; it is merely a title announcing the current page. If it were clickable, it would just reload the current page — not the ideal user experience.

When jets and nature lovers collide

When you combine the jet power of a global navigation system with a bread crumb-style navigation trail, you have a flexible interface that's pretty close to foolproof, and one that's robust enough to handle large-scale sites with multiple levels of content within each section.

Here's one way to use these navigation devices together: Use the global navigation system to house links for your primary sections. This lets people quickly bounce around to find the area that they're looking for. If your primary sections have a lot of subpages, use a drop-down menu to reveal subsection navigation. As people drill down within each section, provide a bread crumb navigation trail to help them stay oriented and able to back out of the section.

Stay in control of your metaphors

A client once asked me to fix a project that was long overdue and way over budget. The project used a 3-D office interface to organize all the navigation choices — choices that had nothing to do with an office, by the way.

The 3-D artwork looked cool, but it took up the entire screen — leaving no room for the actual content except to appear in a layer on top, and therefore covering up the navigation. The interface looked like an actual office — with desks, file cabinets, phone, computer, and so on. All the content choices were mapped on the various objects in the room. Not only did the office look have nothing to do with the content choices, but the 3-D interface was also expensive to design, produce, and update.

User test after user test revealed that no one could correctly associate the navigation choices with the room objects. Even though the client was totally infatuated with the 3-D interface, it just wasn't working for the site. In less than 30 days, we chucked the 3-D interface and designed an elegant, straightforward solution that salvaged the project.

Keeping metaphors under adult supervision

Designers are commonly tempted to use real-world metaphors as a way to creatively organize and navigate Web content, or other interactive content for that matter. In my experience, these kinds of metaphors can get ugly very quickly. They constrain your content and navigation options because you're forced to find things that make sense for the metaphor and not the content.

For example, imagine that your client wants the Web page to look like an open book — complete with curled pages — even though the content of the Web site has nothing to do with literature or books. The book interface automatically locks you into a constrained set of navigation options. The curled pages automatically suggest that users must flip left to right through the content — moving their mouse from one side of the screen to the other just to go back and forth in a linear fashion through the content.

If you take metaphors too seriously, you can end up wasting time trying to match your content and navigation to the theme. For example, staying true to the page metaphor would limit your interaction choices to items that made sense on the page of book, such as text and graphics — but no forms or pull-down menus.

If you decide to use metaphors, stay in control! Don't let them carry you away to the point where they limit you, like the room metaphor shown in Figure 4-15.

Figure 4-15: Can you tell which object in this room interface takes you to the Games section?

5

Web User Interface Design

*I*f you're like most people, you've gotten lost or turned around in a site at some point in your Web surfing career. When you can't see an obvious navigation scheme, or are in the middle of a poorly designed task sequence, it's no wonder you get lost or confused.

As a designer, you have the ultimate responsibility for a Web site's so-called ease of use. After all, graphic design is communication design and is intrinsically tied to the usability of the site. Users can tell a lot about what a graphic or icon does or doesn't do simply by the way it looks and where it's placed relative to other elements on the page. Similarly, the interactive widgets like drop-down menus and check boxes that you select and arrange on the page, and the order you present them in, all affect the success of a user's ability to accomplish tasks within your site.

While Chapter 4 discusses organizing and navigating Web content from a conceptual "information design" point of view, this chapter delves into how people interact with elements on the page and how the visual design can enhance or inhibit usability.

Interaction Design

Interaction design is different from information design. This is often a major point of confusion for people new to Web design. While information design deals with the overall structure of a Web site and the best way to organize content, *interaction design* is the actual flow that users follow to complete tasks such as signing up for a site's newsletter, navigating from one page to the next, or buying a number of products and going through the checkout process.

As a designer, you must think through all the possible steps someone needs to take to complete or abandon a task. You can then design a user interface that facilitates the task, reduces abandonment, and optimizes a user's time spent on your site. In addition to deciding on the appropriate use of widgets like check boxes, menus, links, icons, and buttons, the visual design and physical location of the interface elements can both enhance and detract from the user's ability to navigate successfully through a task.

Creating user flow diagrams

In order to work out the most logical path users should take to complete a task, interaction designers often create *user flow diagrams* like the one shown in Figure 5-1. These diagrams anticipate all the possible paths someone could take through a task. As you can see, each decision junction can go in a couple of directions, oftentimes looping back to a prior step in the task.

After you account for all the possible outcomes, you can figure out how best to design the interaction. For example, you might decide to do the following with your design:

✔ Present the initial set of options in a drop-down menu that updates the page with the appropriate content.

✔ Include a Forgot Password link on the first page of the login sequence.

✔ Include a Register Me check box at the close of an online sales process to encourage registrations because you've already captured more than half the data you need.

✔ Provide users with feedback letting them know they are on step 2 of 5.
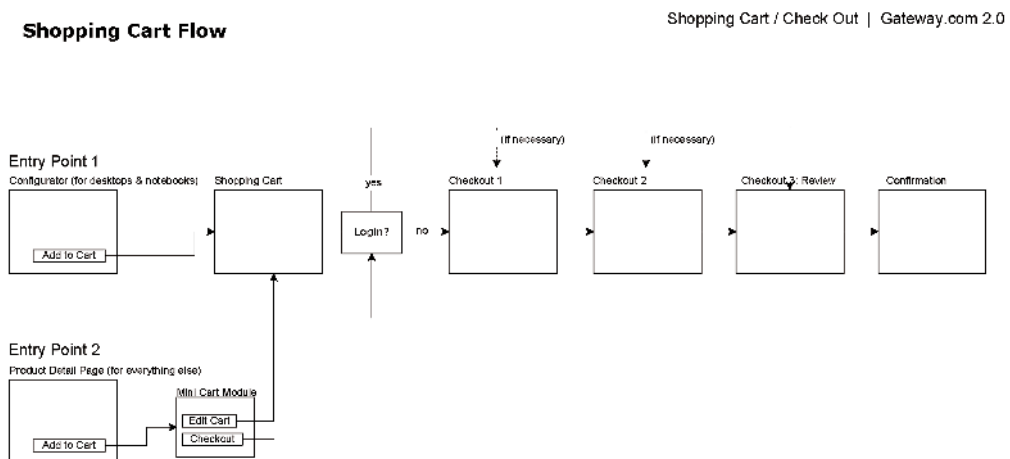
© Gateway

Figure 5-1: A user flow diagram maps all the possible routes a user can take towards completing a task.

Going with the flow

Most tasks are unique to your Web site, requiring you to create user flows and figure out the best interface design to get someone through a task. However, your Web site includes a number of common tasks such as logging in, navigating to the home page, and searching. For these everyday tasks, you can use standard interaction models that have emerged as the Web has matured over the last few years.

Here are a few common conventions to keep intact, or at least understand fully before you tweak them into new variations:

- ✔ **Search function.** If you're providing a search function on your site, keep in mind that most search functions are located on the left side or on the top of the Web page. People look for search functions in these locations so if you place it elsewhere, your visual design really needs to call it out (unless searching is a low priority task for your site).

  Additionally, as Figure 5-2 illustrates, search functions have a common 1-2-3 punch formula: The word *search,* followed by an input field (sometimes with a modifier drop-down menu), and a Go

  Figure 5-2: Place the search function in the top or left top portion of the Web page.

  button. If you use a word other than *search* (such as *find*), or provide just a simple Search text link, people might second-guess the functionality because it's not what they were expecting.

- ✔ **Link to the home page.** For some reason, it has evolved that the company logo on a Web page (usually located in the upper left, less often located in the upper right) is typically the official link to the home page. To accommodate newcomers, however, most sites also provide an explicit Home link as part of their global navigation scheme.

- ✔ **Log in link.** Unlike the search function, most Web sites do not provide text input fields on every page for the login function. Instead, they provide a single link called Login, Log in, or Account somewhere at the top (usually top right) of the page that also doubles as the registration link (even though nothing says registration on the link, although often I see sites that show a combined Login/Register link). The link takes people to a login page where they can either log in with their user names and passwords or sign up to be a new user (with that function falling below the login area).

  This login page often has other features like a Remember Me link, which sets a *cookie* (a wee bit of data stored on your computer) so that users don't need to log in the next time they visit the site and a Forgot Password link for those users who continually forget their passwords.

Visual Design's Role in Usability

Working out the interaction design for a task is one part common sense and one part logic to figure out the widgets to use and the flow to arrange them in. The rest is all visual design. The very placement, grouping, relative arrangement, and design of your interactive components can go a long way toward making a task flow nicely or stop in its tracks.

Compare the two examples in Figures 5-3 and 5-4. Figure 5-3 shows all the same interactive components as Figure 5-4. Figure 5-4, however, is better organized from a visual perspective with added helper text, headlines, and examples of how to fill out the form. The example in Figure 5-4 is clearly easier to use than the example in Figure 5-3. You can use a number of visual design strategies to aid in the usability of your interaction design. The following section offers some ideas to think about.

Figure 5-3: This form example lacks visual design and organization, which makes it confusing to follow.

Yes, I'd like to receive information

PERSONAL INFORMATION
* required

*First name:

*Last name:

Email address: (i.e. name@company.com)

*Phone: (with area code)

*Best time to call: ● Morning
                       ○ Afternoon

ADDITIONAL INFORMATION

I'd like to receive
information about:

I heard about
you through:

I plan to buy:

Submit

Figure 5-4: This example has clearer visual organization, slightly better design, and helper text that makes this form much easier to use.

Giving rollover feedback

One sure way to let people know that things are interactive is to make them change their appearance or trigger some action when the cursor comes into contact with them. When the cursor comes into contact with something, it's called a *rollover*. A number of things can occur when the mouse rolls over an interactive component. If the element has a link, the cursor itself changes from an arrow to a pointing finger. That's a simple, built-in way to provide visual feedback letting people know an element is interactive. However, you can provide additional feedback upon rollover — feedback that can even coach someone how to use the navigation. Here are some ways to do that:

✔ **Change the appearance of the graphic.** You can change its appearance when the user rolls over or clicks it. For example, a normal state for a button might be a certain color, the rollover state might be orange, and the clicked state might be a dark red. Take a look at Figure 5-5. Here you can see the normal, rollover, and clicked state of a single button. Often designers don't bother with providing a clicked state. Remember, each state is a unique graphic you need to produce.

Figure 5-5: A rollover changes appearance when a user rolls over or clicks it.

✔ **Trigger an audio clip to play.** This way is less common to provide rollover and/or click feedback.

✔ **Animate the element.** One interesting way to provide interactive feedback is to make the element morph or move slightly when the mouse rolls over it. The animation can even suggest how the user can ultimately interact with the element. For example, if you roll over something and an animation reveals a dial, you know that if you click, you might activate a dial-like interface.

Designing buttons that look clickable

If you simply rely on rollover feedback, or the cursor changing into a pointing figure upon rollover, you are making the user scrub the cursor all over the screen looking for interactive stuff. Therefore, making interactive elements look clickable to begin with is a good practice.

In general, you can make graphical interface elements look more clickable in one of three ways:

✔ **Use an action word.** Graphic elements (such as icons) without a text label next to them are often disregarded. They either look like page graphics or users can't decipher what they do when clicked, so users avoid them. Use button labels that have a short action word or phrase starting with a verb such as *Print* or *View map*. Passive labels like *Virtual tour* or nebulous marketing-speak labels like *Magical journey* are less effective.

✔ **Add dimension.** Make the elements physically stand out on the page by giving them a three-dimensional appearance.

✔ **Encapsulate the element.** By visually containing a graphic or a graphic and text combination, you set it off from the rest of the page, which implies interaction. Look at the different variations of encapsulation shown in Figure 5-6.

Figure 5-6: By putting an interactive element inside a container, it's set apart from the page design and implies interactivity.

Taking clues from everyday life

The visual design and texture of your interactive components help people know how to interact with them. Objects that you interact with on a daily basis, such as buttons on a phone, drawers, and dials, are great inspirations for the design of interactive elements on your Web page. People are already well-trained to use these everyday objects — everybody knows how to push buttons, pull drawer handles, and turn dials.

WEB SPEAK

For example, if you design an interactive element that looks like a nubby handle, the implication is to click and drag on that spot. In user interface design circles, these visual clues are called *affordances*. Therefore, if your interface graphics borrow the affordances of everyday objects, users have a good idea how to use them.

Take a look at both the dial interface and the drawer scheme on Apple's QuickTime player in Figure 5-7. Figuring out how to use these little widgets to control the volume and access a menu of options is easy. For the volume, you click and drag the dial to turn the audio up or down. For the drawer of options, you grab the handle-like area and drag downward.

Volume dial

Drawer handle
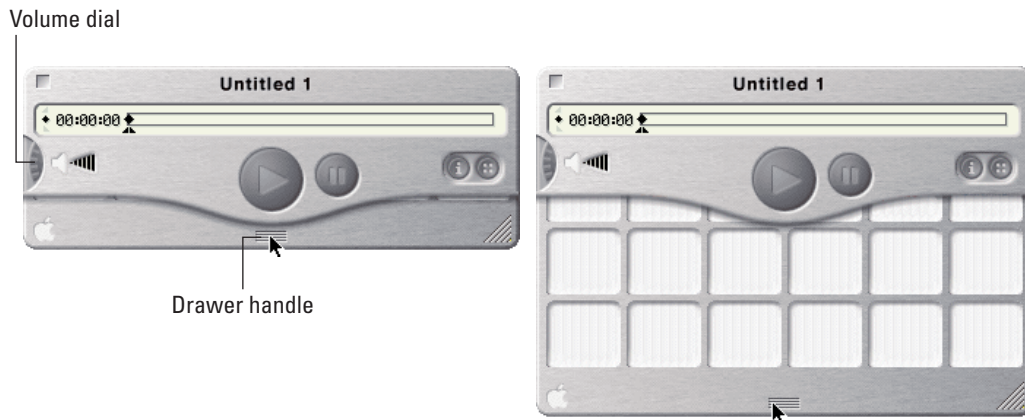
Figure 5-7: The controls on Apple's QuickTime player need no explanation on how to use them.

Grouping and nesting elements

The relative location of interactive components is as strong a clue as any other visual treatment. For example, just grouping a series of similar-looking interface elements together in one graphical unit can give them a clickable appearance — even if the individual graphics themselves don't look clickable.

For example, if you string together a row of plain text labels with the same font treatment, you create a stronger graphical element that stands out from the other things on the page. As shown in Figure 5-8, seeing simple text in a group accomplishes two things:

- ✔ **A group implies interactivity.** A set of text or graphical elements grouped together creates one large visual unit that draws people's attention. The very fact that they are grouped implies something fishy and that the group is probably interactive.

- ✔ **A group implies a similar function.** By visually associating a set of links, you imply that they all perform similar functions. For example, all of the main navigation buttons should look like each other, and rather than being spread out all over the page, they should be contained in one area. If you have a set of less important links, you should still provide a unique visual treatment for them, but group them elsewhere on the page.
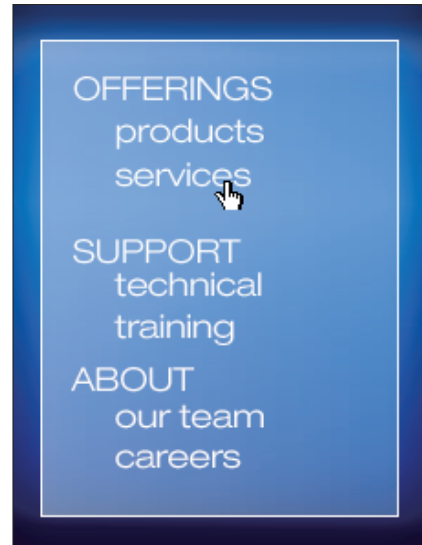
Figure 5-8: By nesting elements under others, you can imply a relationship between them.

You can also imply a relationship within grouped items by *nesting* certain elements under others. For example, take a look at the example in Figure 5-8. Because Services and Products are indented, you know that they are subsections of Offerings and not whole other unrelated sections.

Providing "You are here" feedback

When users browse through sites, they like to have a sense of where they are in the scheme of things, and if they are working through tasks, how long before they will be done. To help answer these questions, your interface can provide "you are here" feedback as follows:

- ✔ **Highlight selected navigation options.** When a user clicks a navigation item that stays visible on-screen (as opposed to one in a drop-down menu), highlight it in some way. Generally, designers create a click state for major buttons. You can keep this click state on to remind users where they are, as shown in Figure 5-9.

- ✔ **Show progress.** If someone is working through a multi-step task, it's a good idea to show a progress meter letting them know where they are and how much more they need to endure. Figure 5-10 shows an example of a text graphic that spells out the steps and highlights the current step.

www.hideawaydressage.com

Figure 5-9: Main navigation schemes often highlight the current selected section so users know where they are in the site.

www.shutterfly.com

Figure 5-10: The graphic at the top shows the user is on step 2 of a 5 step process.

Color-coding

I only mention color-coding because so many people put credence in it. I am not one of them. The only place color-coding works is when you have only a few sections and you want to brand each one (like a conference with three different tracks). With so few sections, users can better associate the color with the respective sections, and the color becomes meaningful. Conversely, if the site has a lot of sections, color-coding falls flat, and people ignore it. They are more concentrated on the sections than what color they are. Case in point: Can you remember the color of Amazon.com's DVD section? Excluded in this debate are sites that use color simply to theme sections and are not expecting to get a usability boost out of it.

In addition, while you can find a few colors that work well together, finding a lot of colors that work well together is difficult. Consider this: When selecting colors for the purposes of branding a number of sections, you need to select colors that all have the same value. Value is the relative lightness and darkness of a color. As demonstrated in Figure 5-11, I have selected three colors that all have the same value and one that is way darker. When I go to use these colors in a site, the fourth color is going to require special handling that isn't consistent with the other colors. For example, I plan to use dark text on top of the first three colors. The fourth color requires light-colored text.

Figure 5-11: The first three colors work well together; the fourth color requires me to treat it differently.

Using icons properly

Rather than creating generic-looking buttons with text, a lot of designers spice up their pages with *icons* — little illustrations that, like pictograms, convey meaning. The problem with icons, however, is that no two people interpret the same meaning from them, so you can't rely exclusively on icons as the main interface buttons. You must attach descriptive labels to them. For a case in point, take a look at Figure 5-12 and see if you can figure out where each icon takes you in the Web site.

Figure 5-12: Can you figure out where each of these icons takes you in the site?

Compare your guesses to where these icons really take you, as illustrated in Figure 5-13. Would you ever have guessed that the clipboard icon is intended to recruit people to join the company? As this example points out, icons can rarely stand alone without some sort of text label to help users know what they do. Except for common functions like printing and saving, designing icons to represent areas of your site is a tricky process. Always couple an icon with a text label, preferably with a short action verb or phrase.

New Products   **Our Catalog**

FAQs   **Join our Team!**

Figure 5-13: Unless properly labeled, icons can be difficult to interpret correctly.

Differentiating between clickable and non-clickable things

Have you ever clicked a button or icon on one page, but when you got to the next page, the same button or icon was a non-clickable decoration? In Figure 5-14, the same graphic that is a button on one page changes to a static element as part of the headline on the next page. This is a big no-no. Remember the old skit on *Saturday Night Live* that featured Chevy Chase hawking a magical product: "It's a dessert topping *and* a floor wax!" In this case, the button is sometimes a "dessert topping" and at other times a "floor wax."

Reusing graphics from page to page to save on download time is smart, but not when it affects usability. The key to a good interface is to visually differentiate the interactive stuff from the content stuff.

When you're designing a site, come up with a consistent visual strategy, or *style guide,* for the various types of clickable things from buttons, to links, to primary, secondary, and tertiary navigation elements. This way, users always know what's clickable and what's not across your entire site.

Figure 5-14: Using the same icon for different purposes confuses form and function.

Consistency Is Everything

When users first come to your site, they look around to get oriented. Hopefully, they quickly absorb your visual and navigation strategy and get on to finding what they came for. After people initially figure out the interface, they don't want to waste their time futzing with it again — the interface should become transparent to them.

TIP

The best way to make an interface familiar and transparent to the user is to be consistent. Once you establish your visual language for clickable and non-clickable things and decide on the location for your interactive elements, follow it to the letter everywhere in the site.

As Figure 5-15 illustrates, changing your visual and or placement strategy for interactive elements within the site not only confuses people, but also makes them think button functions have changed — even if they retain the same name.

First, it's a button...   ... now, it's a text link

Figure 5-15: If you change the look of a button, the user may think it's a new button with a new function.

REMEMBER

Believe it or not, the one situation that you can get away with a change of navigation placement and visual treatment is the home page. Oddly, a number of high profile Web sites, like the example in Figure 5-16, introduce the main navigation front and center with an expanded visual treatment on the home page. Then, when you click to go to a subpage, as shown in Figure 5-17, the main navigation shrinks and finds a happy home somewhere else on the page (usually in the upper left or upper right) where it remains throughout the site.

The only way this strategy can work is to ensure that the visual design of both the home page and the subpage navigation feels like they are related. Use the same font treatment (perhaps only changing the point size), use the same background and borders for your buttons (just change the size), or maybe add some helper text and an image next to each navigation choice that disappears on the subpages.

I'm not personally a fan of this strategy because I still think it's risky to change the main navigation around. Not to mention, your development team won't be happy if they have to build two versions of the so-called "global" navigation.
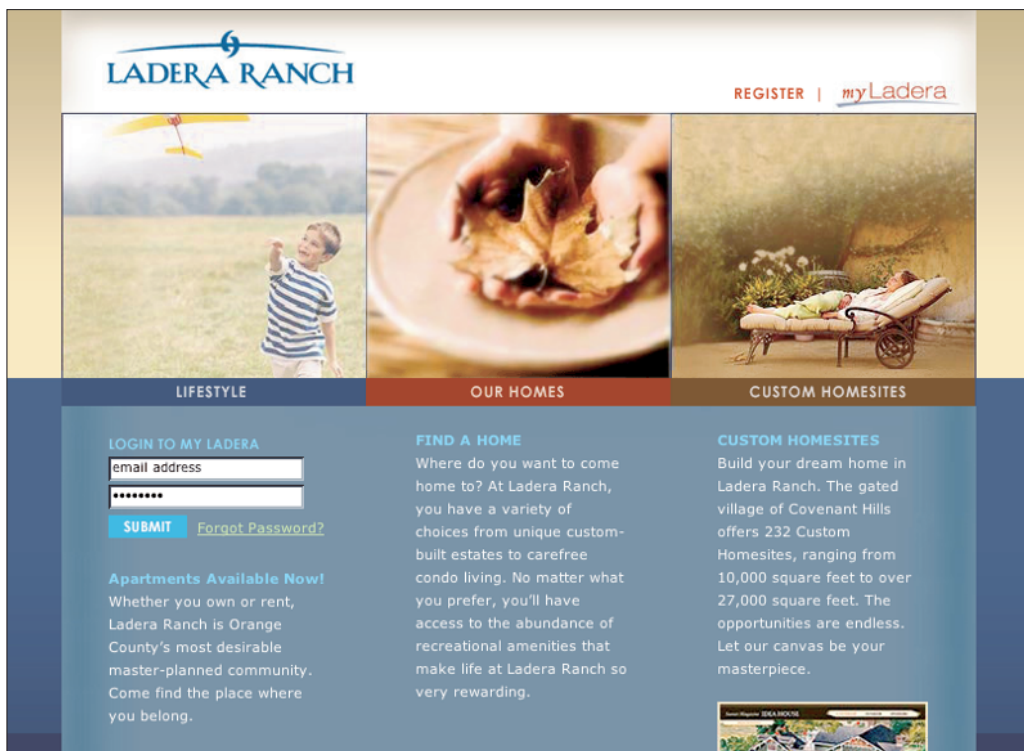
www.laderaranch.com

Figure 5-16: This home page introduces the main navigation in the middle of the page.

Figure 5-17: When you click off the home page, the main navigation shrinks and goes up to the top where it stays throughout the rest of the site.

Alternative Interactive Design Strategies

All the strategies that I discuss in previous sections of this chapter to make an object look interactive have one thing in common: They all provide some kind of *visual* feedback to the user that screams "click me please." Other kinds of feedback, however, can be used to give people clues that an object is interactive. You can also use sound, animation, and media to enhance the usability of your interface components.

The following is a list of some tried-and-true techniques that make people stop and think, "Hey, there's something funny about that graphic":

- **Audio.** One way to make an interface element stand out is to use audio. Like a rollover button, you can play an audio clip when the user rolls over the button and a different audio clip when the user clicks the button.

- **Animation.** Nothing grabs a user's attention like movement on a Web page. Until recently, most Web pages were fairly static and silent, so any movement on the page stuck out like a sore thumb. These days, animation is more common than it used to be, but it can still command attention when used correctly.

The key is to use animation sparingly and to concentrate it on one area of the page. If the whole page is twinkling and twirling, forget it. Use animation only for important interactive elements. For example, make the lead story or featured product a Flash movie. You can also make your main interface buttons animated, or animated only when the user's mouse pointer rolls over them.

Maximizing Space

Because space is at such a premium on a Web page, the smaller your interface elements are, the better: You can fit lots of small elements on a page. However, the smaller you make interface elements, the less usable they become: Your visitors will need a spyglass to get through your site.

So how do you get the best of both worlds and maximize your Web page space? With some clever use of Web technologies, you can design small interface elements that expand when the user rolls over them to present more options. Here's a catalog of different techniques:

✔ **Disjoint rollovers.** Similar to a rollover button, a *disjoint rollover* changes the appearance of a graphic on mouse rollover, but the change occurs elsewhere on the page. For example, if the mouse pointer rolls over a small button, a larger adjacent area changes to reveal more information.

Take a look at the rollover in Figure 15-18. When a user rolls the mouse pointer over the button, a large area appears next to it with more information. The additional information can help the user decide whether to click.
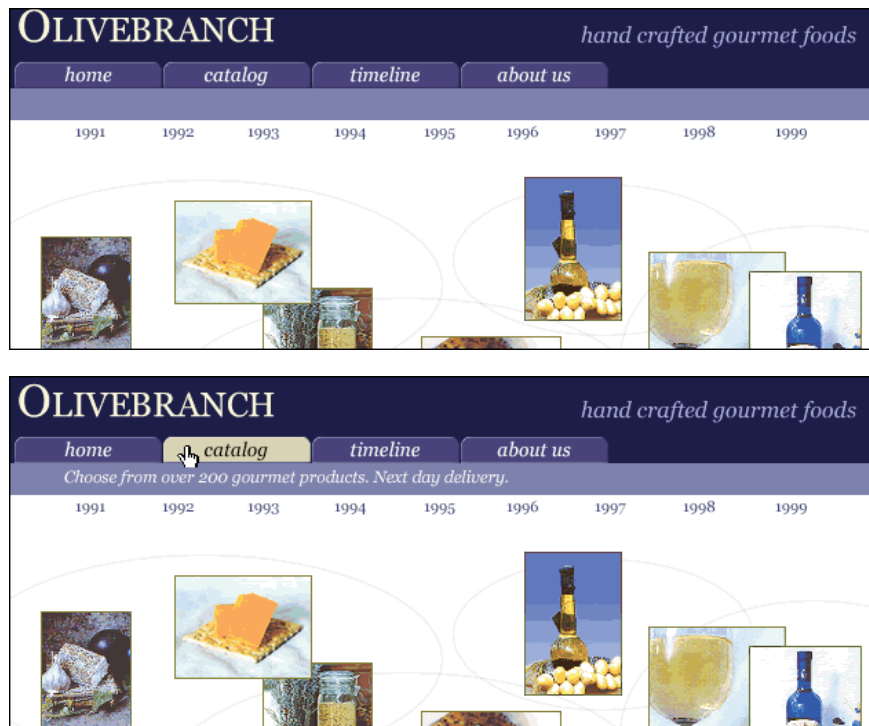
Figure 15-18: When a pointer rolls over a button, a larger area appears with more information.

You can really use disjoint rollovers to their full advantage by using them not just as extra info for the button, but to display page content. Take a look at Figure 15-19. As you roll over each of the tiny thumbnails, the whole screen updates with new information.

Figure 15-19: You can even use disjoint rollovers to display additional information.

✔ **Drop-down menus.** Another way to maximize your space is to use *drop-down menus.* A list of additional navigation choices appear when the mouse rolls over a button. The concept is similar to a disjoint rollover; however, the content of disjoint rollovers is not interactive. Drop-down menus give users a more refined set of choices so they can jump quickly to the area that they need. You can also use drop-down menus to consolidate a lot of your navigation choices so you have less stuff on the screen, as shown in Figure 15-20.

A button style drop-down menu   An elegant, graphical style drop-down menu
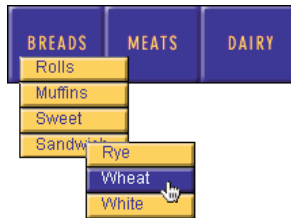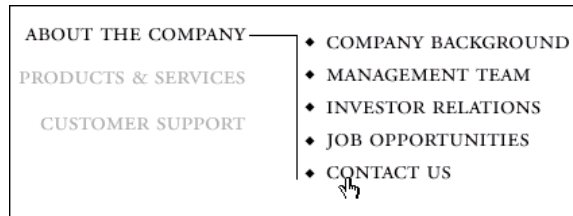
Figure 15-20: Drop-down menus reveal more navigation choices when the mouse rolls over a button.

✔ **Shrink and expand.** As the title implies, you can design content to expand upon rollover much like the Macintosh dock expands in a "fish-eye" effect as you roll over the icons.

As shown in Figure 15-21, when the mouse rolls over an area, it expands outward in an animation to reveal a full range of navigation choices. This example is the excellent mummies feature at www.discovery.com. This elegant way conserves space and gives users full-service functionality and an immersive experience all at the same time.

Photo courtesy Second Story

Figure 15-21: This interface starts out small and then expands when the mouse rolls over it.

6

User Testing: Lab Coats Not Required

In This Chapter

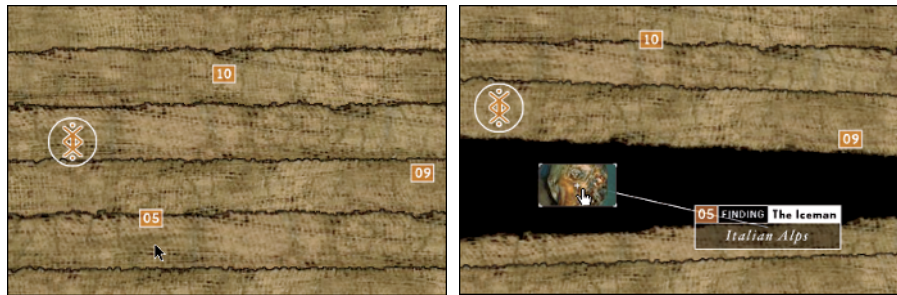▶ Creating clickable wireframes to test

▶ Focus group testing your visual designs

▶ Prototyping key task flows

▶ Recruiting users for a test

▶ Conducting user tests

▶ Evaluating test results

*A*t some point in the design process, you have to put your work in front of a group of perfect strangers for the ultimate litmus test: Can they figure out how to use it? And, do they like how it looks?

Ideally, you should test your work as early in the design process as possible in order to stave off any unforeseen usability problems before you're knee-deep in production (which is too late). By putting together a workable prototype early in the design phase, you can organize a bona fide user test, complete with a list of questions to ask and a plan for recording and evaluating the feedback.

Although a user test sounds academic and may conjure up visions of white lab coats, it's actually a fun and truly enlightening experience. You'll be surprised at the things a user test reveals about your design, both from a visual standpoint and from a usability standpoint. In this chapter, I discuss how to make prototypes that you can put in front of users and how to get the most out of your testing efforts. In the end, you'll be glad you took these extra steps. You'll end up with a Web site that is not only beautiful, but works too!

Developing Testable Prototypes

Before you can gather any useful feedback, you have to assemble a prototype that you can test. Working with a sketch on paper isn't good enough; people need to see and touch the real deal — HTML pages working in a browser.

REMEMBER

I have found it useful to create two sets of testable prototypes during the design phase — clickable wireframes to test your information design, and HTML click-throughs to test the usability of your visual designs. You should share both prototypes not only with end users but also with your clients so they can get a feel for, and can approve, the end product before it's built.

Creating clickable wireframes

Early in the process, clients need to understand how the site is organized. In my experience, for large or complex sites, just looking at blueprint-like diagrams on paper is not enough for them to visualize the interaction and organization of a site. So, after you work out the site map and begin wireframing, set aside some time and money to assemble a working wireframe edition of the site in HTML. (***Note:*** For small sites that are 40 pages or less, walking clients through a PDF of your wireframes is plenty.) For more on site maps, see Chapter 3. For detail on wireframes, check out Chapter 4.

Most designers and information architects use Microsoft Visio or illustration programs such as Macromedia Freehand or Adobe Illustrator to create wireframes. All three of these programs have the ability to export wireframes as GIFs and JPEGs, which you can then insert in HTML with a program like Macromedia Dreamweaver.

I recommend exporting wireframes as GIF files. Because they are (should be) grayscale and text, you get better quality and compression saving as GIFs.

In Dreamweaver, insert the whole GIF wireframe into the page, just as you would insert an image. As shown in Figure 6-1, you can then draw *hotspots* (clickable regions that you define) on top of your navigation and link them up to go to your various pages. Voilà — a clickable wireframe prototype ready to share with your clients.

TECHNICAL STUFF

If you are handy in HTML, you can take your prototype one step further by building a working navigation system separately and using CSS (Cascading Style Sheets) to plop it into a `<div>` layer on top of your wireframe image. See Chapter 13 for more on creating `<div>` layers.
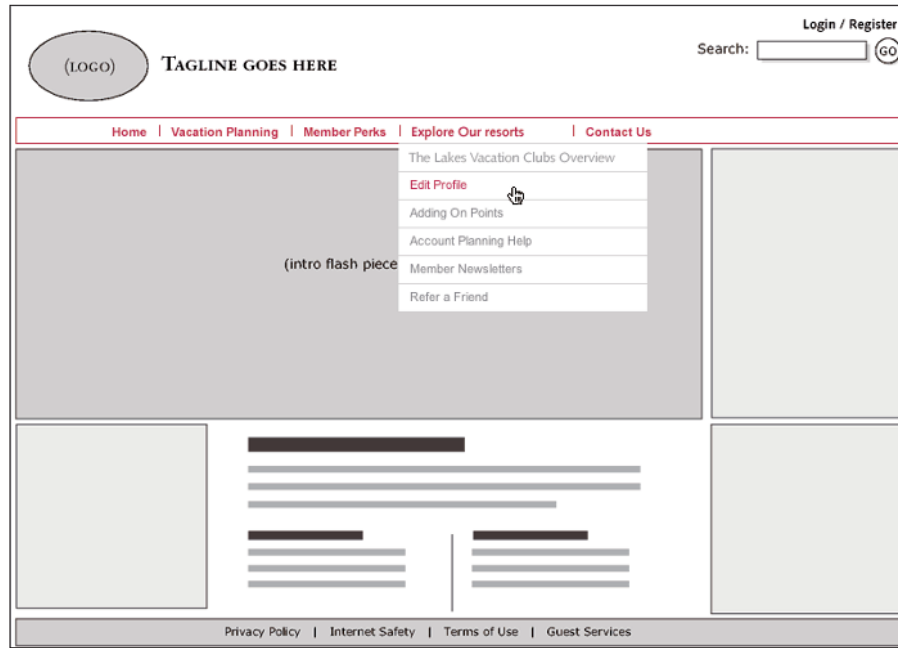
Figure 6-1: Use a program like Dreamweaver to create a prototype site you can test with clients.

Present your clickable wireframe edition to the clients as a way to help them visualize how the site is organized and how users would navigate through key tasks such as ordering a product. You can also present the clickable wireframes to end users to see if the site makes sense from their perspective. Later in this chapter, I discuss ways to conduct tests with users. At this stage, it's important to work out all the kinks in the site and get client approval before moving forward with visual and HTML production.

Testing your visual design

While the wireframe details are being worked out, the visual design team can create a few different "look and feel" options and test them with users. Aptly named "look and feel," these designs look real enough, as in Figure 6-2, to pass for a home page and an interior, or subpage, of your site, but they are not 100 percent accurate in terms of final navigation, photography, or text. However, keep in mind that users and clients are very literal. Selecting head-line copy and photography that are fairly accurate to get a good read on your audience is important. All non-major text should be *greek* (gibberish text), but navigation choices and key headlines should be a close approximation.
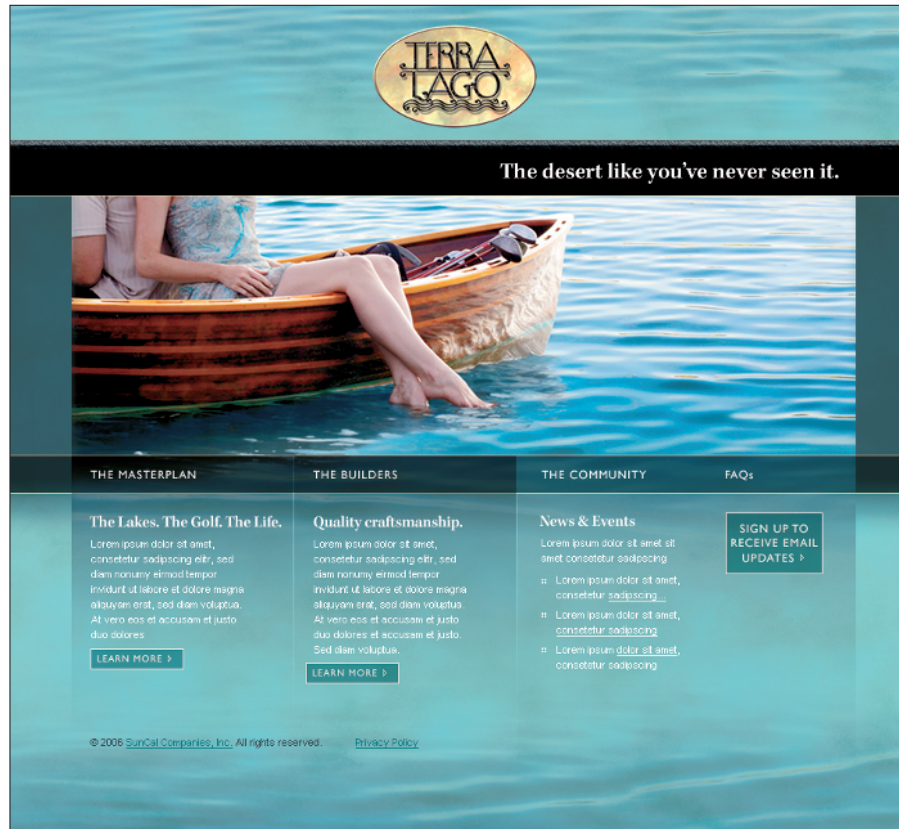
www.terralago.com

Figure 6-2: This design direction was one of three shown to the client and was ultimately selected. Notice the greek text.

For high-profile Web sites, it's a good idea to conduct a focus group before committing to a design direction. Otherwise, the design is chosen based on the personal tastes of a few client executives — which is okay if the site is for a small to mid-sized company. I like to get focus group feedback before showing the design to the client. That way, you can rule out designs that just don't work at all, and come prepared with user feedback when you go into the client creative presentation to help guide the decision making process.

To test your visual designs, follow these steps to put together a focus group:

1. **Recruit seven to ten people that represent your intended audience.**

2. **Print each of the design options.**

   I like to present three different design sets of a home page and a subpage.

3. **Mount all the designs on boards so that your focus group can get a good look at each one, refer to them, and compare them side by side.**

4. **Prepare JPEG versions of your designs (saved at 100% quality for the purposes of demonstration) and place them in HTML pages using Dreamweaver.**

   This enables you to show the focus group how each design looks on a computer screen in a browser window.

5. **Gauge people's emotional reaction to the designs.**

   See which design most resonates given the type of business your site reflects. Sometimes there are clear winners and losers. If all rate about the same, then truthfully any of the designs can probably fare just fine in the real world.

Building an HTML click-through to test for usability

Not only do users respond to the mood created by a site's visual look and feel, but as Chapter 5 discusses, the visual design also has a big impact on how users identify interactive elements and navigate through a site. So, it's a good idea to mock up a few pages in the chosen look and feel that illustrate a user's path through a key task such as ordering a product.

Cruise line case study

For a major vacation cruise line company (with ships bigger than the Titanic!), we created each wireframe in Dreamweaver. Because each wireframe was already in HTML, the prototype site virtually built itself as we created each new wireframe page. While the IAs (information architects) took some time to get comfortable building wireframes in Dreamweaver (as opposed to Visio), the time was well made up during the client approval stages. Each week we'd walk the client through a different section of the site — the online booking process and the reservation retrieval process for example. The client sessions were highly productive because, for such a large and complex site, the client could really navigate through the site and provide accurate feedback and faster approvals.

Assemble these few key pages into an HTML *click-through*. As shown in Figures 6-3 and 6-4, the click-through looks real and contains all the pages that users would see if they were to step through a task successfully. Each of the pages can be a mock up saved as a single JPEG and placed in an HTML page. In fact, you can expand upon your initial design directions to make your first testing prototype.

WARNING!

For testing purposes, don't mock up only half the page and leave the rest blank. During the test, users will ask about these neglected areas. Even if you give them an explanation, they are forced to use their imagination to fill in the gaps. This can skew their reading of the page. Users need to see the page in its full context to give you the most accurate feedback.

REMEMBER

For testing purposes, the HTML coding for each click-through page should be just enough to place the mock up image in the page and apply a few hotspots to interactive areas that link to the next page in the task sequence. At this point, you just want to see if the visual and interaction design work as you intended.

Figure 6-3: On this home page, it's pretty clear where you click to upload your photos.

© Shutterfly, Inc.

Figure 6-4: The first step is to register by entering basic information.

On testing day, you may consider having a few design options on hand for the users. As you begin to ask questions, you may present alternative ways of visualizing the same page with different button treatments and ask users to compare which way they prefer.

During the testing phase, users may offer suggestions on how the interface can be better or different. After all, the whole point of user testing is to open up to new ideas that better serve the end user. If you've already invested a lot of design time to make the site work a certain way, you may be reluctant to change it.

Prepping for the User Test

The most efficient way to conduct a user test is to send people on a kind of scavenger hunt to perform certain tasks. Prepare a list of two to five tasks that you want to test and then create a series of click-throughs that provide alternate ways to do each task. Keep in mind that the click-throughs are *your* ideas about how the tasks should be done. In a user test, it's always interesting to see which path a user takes within each path (hence the reason you hook up a couple alternative options in your click-through).

TIP

Along with your list of tasks, you should start thinking about what you want to find out from testing each task. Make a separate list of your own questions. This helps you to standardize a testing methodology that you can use on multiple people.

The more people you test, the more accurate the feedback you get. For example, if only one person out of five doesn't like the button design, you can probably chalk it up to their own subjective taste. If that person was the only person you tested, you could end up redesigning something that was fine in the first place.

Preparing to-do lists for users

REMEMBER

In preparing a list of tasks for the user test, keep in mind the main goals of the Web site. For example, if the main objectives of the site are to get people to buy products and to become registered users, these goals should drive the list of tasks to test.

Like the example shown in Figure 6-5, create a task list document that you can give to each user. Make a separate edition for yourself that includes your own questions for users about each task. In the following example, I assume that the goals of the site are "First, get users to buy products. Second, sign up to be a registered user." Given these objectives, here's how to make a task list:

1. **Create a scenario that places the user in a role.**

   For example, tell them, "You're a customer looking to buy a book and sign up to be a registered user because you've heard you'll save 10 percent on your purchase." As you can see, this scenario exactly mirrors the top two goals of the site, but is a little more focused on a typical task.

   Place this scenario statement at the top of the document that you give to users. With the proper mindset, users' comments become more focused and relevant. For example, they may say, "Well, if I hear that I can get 10 percent off by registering, I'd want to register first." This is a great nugget of information! Because you want them to buy first, not register first, the designs should funnel people to the checkout pages. The checkout pages should then provide a clear option to register and save as part of the checkout process itself.

2. **Create three to four mini tasks that you want the user to try.**

   List these tasks below the scenario statement. The tasks can either be in the form of questions, such as "How would you buy a book called

Web Design For Dummies?," or action requests, such as "Tell me how you would sign up to be a registered user." The questions should be targeted toward a specific action and not too broad, such as "How would you buy any ol' book?"

3. **Make your own cheat sheet.**

Create a separate document for yourself that lists each scenario and task set just as it does on the user's copy. At the top of your cheat sheet, however, list the main objectives of the site. Like the example in Figure 6-6 shows, within each set of tasks, write your own questions so you can remember to ask the same things of each user.

User scenario:
You're a customer looking to buy a book and register because you've heard you'll save 10% on your purchase.

Task 1:
In looking at this interface, where would you start to buy Web Design For Dummies?

Task 2:
How would you add a second product to your order, say a music CD?

Task 3:
How would you get to the check out area?

Task 4:
How would you sign up to be a registered user?

Any additional comments or suggestions?

Figure 6-5: A sample user testing sheet.

Goal:
Sell products and generate sales leads (via registration).

User scenario:
You're a customer looking to buy a book and register because you've heard you'll save 10% on your purchase.

Task 1:
In looking at this interface, where would you start to buy Web Design For Dummies?
• Can the user find the books section of the store?
• Can the user find and use the search function?
• Is the navigation clear?
• Does the user understand how to add the book to an order?

Task 2:
How would you add a second product to your order, say a music CD?
• After the user adds the book to the shopping cart, can he or she find the other product sections?
• Does the user think their initial book order will be lost if they continue shopping?

Task 3:
How would you get to the checkout area?
• Does the user find the checkout area?
• Does the user know how to continue shopping even while in the checkout process?
• Is the ordering process clear?
• What sort of order confirmation sequence does the user expect?

Task 4:
How would you sign up to be a registered user?
• Does the user see how to register?
• After registered, can the user see where to log in upon returning to the site?
• Does the user understand or care about the benefits of registering?

Any additional comments or suggestions?

Figure 6-6: Create your own "cheat sheet" version of the testing sheet.

Developing a testing methodology

After you develop a list of tasks for the user and your own cheat sheet (as described in the preceding section), the next step is to come up with a standard testing methodology. By standardizing one testing methodology, you have a more accurate means of evaluating the test results. In addition, you can delegate the testing to a few people on the team.

Here's a checklist to help you build your own testing methodology:

- **Target audience.** Before you recruit a bunch of people to be your test subjects, you must know what kind of people to go after. Recruit people who fit the Persona profile you developed for the site (as Chapter 3 discusses). For example, a shopping site with books, music, and videos may target Internet-savvy 20- to 35-year-old professionals with money, but not time, to burn.

- **Number of users.** The more users you test, the better your results. Testing more than a few people is critical because you can minimize the effects of each individual's bias.

  For example, if only one person out of six complains about the look or placement of a certain button, you can probably disregard the comment. If five out of the six people have the same complaint, you know you have a genuine problem. A good number of people to shoot for is seven to ten people. You should also make sure that you have a good mix of men and women (unless the site focuses on one group).

- **Testing style.** Because you can test in more than one way, you should decide which style of testing you want to use. The serious lab coat style of testing involves setting people up in a room by themselves behind a one-way mirror (I kid you not) and videotaping their actions as they work through the task list. I've found that this method is way too intimidating for users — they don't want to look stupid on camera, so they hold back.

  The other, more casual way, is to sit down with them side by side, asking questions and taking notes as they respond. This method is more empowering, and users really seem to spill their guts and tell it like it is. You can still videotape the session. Videotaping enables you to see where the mouse goes on the screen as a user works through the tasks, and record any feedback in case you can't keep up with notes.

Carrying Out the User Test

The whole point of user testing is to catch the usability problems of your design before you go into production. Therefore, you must be mentally prepared for the feedback that comes out — it may not be pretty. In some cases, you may be faced with a substantial redesign if user after user can't figure out your interface.

Give yourself a good two weeks before testing day to line up enough people to make the test worthwhile. Recruiting people to donate their time for an hour or two can take some doing. To help recruit them, tell them exactly how long the test takes — hopefully no more than one hour — and let them know what sort of compensation you can offer.

After you get people in the door, make sure you honor the time commitment that you promised and stick to your testing plan. Other than that, just sit back and take notes!

Finding willing guinea pigs

The real trick in user testing is to actually find and convince a bunch of people to be willing guinea pigs for your test. If your target audience is 25- to 40-year-old busy professionals, you may have a harder time recruiting people away from their jobs than if your audience doesn't work or has flexible work hours.

If your client has an established clientele, coming up with a list of people to call on is fairly easy. Simply cull its customer database for people in your local area (unless you are conducting tests in a few cities). The best approach is to call these people directly. I've found that in this impersonal online world, people respond warmly to a personal phone call — especially when they are made to feel that their opinions count.

For new Web sites that have no clientele, finding a good group of testers is a little more difficult. In this case, the easiest path is to recruit friends, colleagues, and family members that fit the profile of the proposed target customer. Have the clients provide you with a list of people that they know.

The other option is to enlist a professional testing company such as Audience Profiler (`www.audienceprofiler.com`). It can recruit people that fit your profile, provide a testing environment, run the test, and provide you with the results.

Friends and family may be more lenient in their feedback because they don't want to hurt your feelings. At the start of the testing, remind them that the only way they can help is to be honest and forthcoming with their feedback.

If the site's clientele is highly specialized, such as doctors and their staff, try calling their offices directly. Getting people to participate in early user tests is also a great way to cultivate future customers.

After you identify a list of testers, you have to sell them on the idea of taking time out of their day to come test. Here are a few tactics that I've found to be successful:

- ✔ **Set a time limit.** Put a time limit on the user test so people know what they're committing to, and schedule a time that's convenient for them. For example, tell a potential tester that the test portion only takes one hour, plus an extra half hour for setup and conclusion. If they come in at 1:30 in the afternoon, they can be out by 3 p.m.

  Make sure you follow through with the promised schedule. If you don't finish all the tasks in one hour, it's your loss. Conclude the test and thank them for their time. If you're almost finished, however, the user usually offers to stay.

- ✔ **Give out swag.** In appreciation of their participation, offer small gifts at the end of the test. Most companies have marketing T-shirts, hats, coffee mugs, and pens with the company's logo, which make great give-aways. In the industry, people commonly call these things *tchotchkes* or *swag*. It's amazing how far these little gifts go — people can never seem to get enough of them.

  In addition to tchotchkes, if your client makes a product such as software, you're sitting on a goldmine of give-aways. There's nothing like giving away a shrink-wrapped product that retails for a good amount to lure potential testers — especially when you consider that it costs the company only a few dollars to make.

- ✔ **Pay people.** Depending on your client's "coolness" factor, you may or may not have to pay people. If the client is a new company that no one has heard of, you may have to offer testers a small cash stipend, such as $20 for an hour of their time. Money should be the last thing in your sales arsenal. Before you resort to giving away cash, see if people agree to come in simply by scheduling a convenient time and offering free gifts.

- ✔ **Send out thank-you cards.** After the test, send users a thank-you card in the mail. Although this isn't a sales tactic to sway people to come in, it's a great way to build good faith for the future. You may be able to call on these people again, or use them as references to find other testers.

Conducting the test

WARNING!

Before you sit down with users in front of a computer, you need to get them to sign a release form and let them know you're videotaping the session. If you are testing with minors, you need to check state laws on videotaping them.

At the start of the test, review the scenario and the task list and describe how you plan to conduct the test. Tell them that . . .

✔ You'll be asking a few questions about each task, and taking notes.

✔ They should talk out loud about their thoughts and reactions as they work through a task.

✔ They can criticize the design or express doubts, such as "I'm not sure what that button does or where it will take me."

✔ If they can't do something, it's probably the design's fault and the reason they are here today. So they shouldn't feel stupid about not completing any of the tasks.

✔ They're looking at a semi-functional prototype, and not all the functionality is hooked up yet.

Start on the first page of your HTML click-through and ask the user to complete the first task. Then, sit back and start taking notes as they talk out loud about what they're thinking. You should hear things like "I'm looking for a Search field. I should be able to just type what I want to find and click a button." Give them some time to think through the problem before you ask any questions about the task.

WARNING!

Avoid asking leading questions that give clues, such as "Do you think that the Search button looks clickable?" Not only are you pointing out a button that users should find on their own, but you're also expressing doubt about the button's design — thus skewing their perception of it.

Because your click-through storyboard is limited in its functionality and only illustrates a couple paths through each task, you must moderate each click. As soon as users tell you where they would click, ask them what kind of page they expect to see next. If they choose the correct button, let them click and then gauge their reaction as they view the next page.

If they choose a path that's not illustrated in your click-through, they may have just given you a great alternative way of navigating. On the other hand, maybe your interface isn't clear. Show them the path that you're proposing

and ask them what they think. They may yield and tell you "Oh, that makes more sense, I just didn't see the button," or they may say, "That makes no sense at all." If their response is the former, you just need to redesign the button or put it in a different place. If they say the design makes no sense at all, ask them if they can think of a better way to perform the task.

After the user completes a task, or you notice that the user is stuck, ask the bulk of your remaining questions and then ask for opinions on the best ways to do the task. After you wrap up the task and your questions, move on to the next task. Each task should take about 15 minutes to complete — any longer than that and you risk burning testers out.

"Houston, We Have a Problem . . .": Evaluating Results

After a day of spending an hour or so with four to six people — running each one through the same set of tasks and questions — you should have many pages of notes to evaluate. The best way to sift through and make sense of all this data is to transcribe your notes into some sort of visual graph.

Consolidate all the feedback for each task into one table, as shown in Figure 6-7. This table shows how you can organize the feedback from testing the click-through in Figure 6-3. A table is a great way to compare all the notes for each task so you can better see where the problems lie. Make a grid that lists each user down one axis and each of your questions along the other axis. Also, leave one column for the users' suggestions and comments. At the top of the document, include the site's goals, the user scenario, and the testing methodology that you used so that you and the client have a better basis from which to analyze the feedback.

With this sort of visual arrangement, you can easily see where the big problems lie and evaluate what things really need fixing. By using a spreadsheet, you'll notice that the comment column often fills up with the same suggestions for fixing the big problems. This makes your redesign process that much easier.

Armed with this data, you can create a new set of designs. You may even consider e-mailing the updated designs to the testing group for one last chance at feedback from them before you go into final production. This also gives your testers a chance to see that their feedback made a difference.

Site Goals:
Sell products and generate sales leads via registrations.

User Scenario:
You're a customer looking to buy a book and sign up to be a registered user because you've heard you'll save 10% on your purchase.

Testing Method:
3 subjects aged 25–40, all professionals with incomes over 60K per year.
One-on-one testing, 1 hour video taped session
On-screen click through shown Web browser

Task 1: How would you buy a book called Web Design For Dummies?

	Can the user find the books section of the store?	Are the navigation buttons clear?	Can the user find and use the search function?	Does the user understand how to add the book to an order?	Additional Comments and Suggestions?
User 1: Male aged 32	Yes, store sections are clearly marked.	Yes, likes consistent mini tab for all buttons.	Yes, though not sure why it's different color or upside down.	Yes, once I found the book, it was easy to use the mini tab button to add the book to my order.	Thinks the search function should always be present on the page.
User 2: Female aged 25	Yes, likes button treatment	All the buttons except for the subnav buttons under main tabs. Not clear that you are in the Featured Products section.	Did not realize that it was clickable. It looks different than the rest of the buttons.	Yes, very clear.	Make the subnav buttons look like the other tabs or like mini buttons. Try a different highlighting system.
User 3: Female aged 36	Yes, easy to find	Not sure about the subnav treatment under the main tabs. Didn't realize that they were in the Featured Products section.	Found it ok, just does not like it's treatment. When you click the other tabs, you go to a page. When you click the Search tab, a mini search window appears. Thinks everything should be consistent.	Yes, likes the way that all the mini buttons look the same.	Feels strongly that the Search button should not be an upside down tab– maybe have the mini search function always visible on the page.

Also, try a different way of highlighting the subnav – make it look more like the main tabs? |

Figure 6-7: User test results arranged in a table help you see what worked and what didn't.

Part III
Designing Web Graphics

The 5th Wave — By Rich Tennant

HORNER BROS.
MAKERS OF PREMIUM
BELLS & WHISTLES

"As a Web site designer I never thought I'd say this, but I don't think your site has enough bells and whistles."

As a creative, soon-to-be Web professional, this is the part that you've been waiting for — the part where you get to flex some creative muscle and start building Web graphics from the ground up. If you've been a print designer or if you're new to the Web and need to know the tools, tips, issues, and techniques for designing Web graphics, this part is for you.

Chapters 7 and 8 walk you through the basic principles of graphic design and typography and discuss how they relate to the Web. As a Web designer, you must also be aware of the technical issues surrounding color palettes, image resolution, and file formats. Chapters 9 and 10 give you an in-depth understanding of these issues, as well as give you hands-on practice creating Web graphics with some of today's best software tools.

In Chapter 11, I show you how to put all your skills together to create a set of design options for a Web site and present them to a client. After the client approves a design direction, Chapter 12 handles prepping final production files and producing Web-ready graphics.
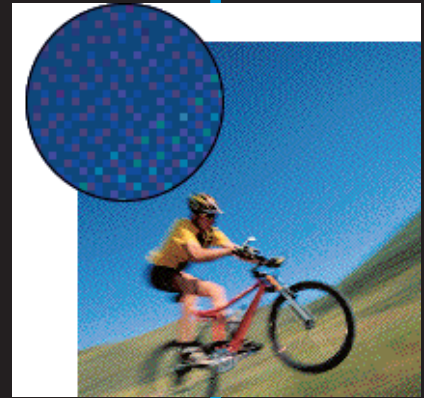
Web Graphic Design 101

The principles of balanced layout and use of text and graphics apply to designing Web sites just as they do in media such as print and video, but a different set of constraints exists. In the Web design world, you have to think small and efficient, readable and interactive. Plus, all sorts of pesky technical things can get in the way of your creativity.

If you're not a graphic designer, this chapter is really for you. I discuss basic graphic design principles of how to blend color, text, and images into a pleasing Web user interface that people don't mind coming back to. If you're a designer already, this chapter covers some familiar design territory, but also shows how it all applies to Web site design.

Crafting the Visual Interface

The visual design of each Web page should accomplish two things: Make the page look professional and, more importantly, show people how to navigate around the site. These are big responsibilities for a humble graphic interface. As I discuss in Chapter 5, the way you design buttons — even text links — and where you place them can make or break a Web page.

WARNING!
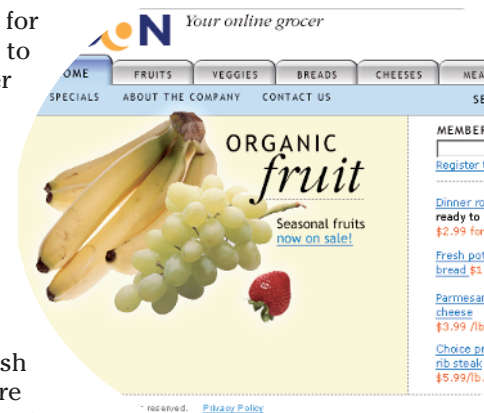
If visitors can't find your site navigation elements because nothing on the page looks clickable, elements take forever to download, or they are placed "below the fold," people will get confused and your site will not be very successful. Notice the difference between the two buttons shown in Figure 7-1. The button on the top looks like a mere heading, but the button on the bottom looks like a 3-D bubble that screams "click me."

REMEMBER

Aside from designing the interactive components to look interactive, you also need to be mindful of where you place them in the layout. If you change the location of a button from one page to the next, the user won't be able to find it. The visual layout, therefore, must remain consistent from page to page and yet accommodate all the different kinds of content in your site.

Figure 7-1: Although both are buttons, only the 3-D one looks clickable.

Developing a page design strategy

Generally speaking, the home page has more layout freedom than the rest of the pages. Each page beyond the home page, however, has a much more limited design layout because you have to keep the interface components consistently placed and yet allow for all kinds of content the site might contain. In some ways, this makes your job easy — you need only to design a home page layout and a subpage layout for your navigation, and carve out a flexible content area that holds different types of page content.

WEB SPEAK

Take a look at the page design shown in Figure 7-2. This design has a simple framework across the top and left side for the navigation. The middle portion, or *content area,* can easily handle a bunch of different elements from forms to text and graphics.

Keeping such a page strategy in mind is important when you begin to develop your visual designs. Otherwise, your designs may end up looking cool, but are completely impractical for Web application.

Creating comps

Comps, as they are called in Web design circles, are mock-ups of the proposed design. Although your initial focus is designing just two main layouts — the home page and the subpage — a lot of design work goes into creating these two pages.
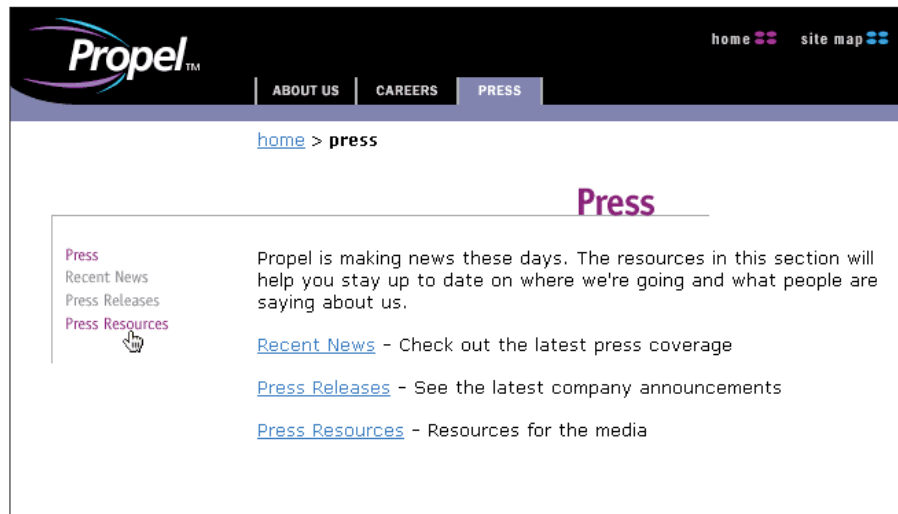
Figure 7-2: The subpage design of this site is a framework that can work for many types of interior layouts.

To create an effective Web page, you have to follow basic design principles for blending color, type, and graphics. You must also use your page space efficiently to achieve a layout that's not too crowded and is flexible enough to display a lot of different media types, such as movies, text, and forms that users would fill out. After you work out the design and layout of these two pages, you can use them later as a guide to build other production comps that handle all the nuances of the site.

Blending color, type, and graphics

Color, type, and graphics are the three main ingredients that go into your Web design. Overdoing any one of these three can ruin the design, making it look amateurish. The next sections discuss Web Graphic Design 101 guidelines for using color, type, and graphics effectively.

Choose colors that are appropriate to the subject matter

For example, if the Web site is selling high-end executive homes, a palette of muted, classic colors might work better than bright pink and green. This is just a guideline of course. It may be that the site is selling modern architectural marvels that call for a clean design with lots of white and some unexpected colors like "mint" and "coral." A good way to start is to think through a list of adjectives that a customer would use to describe the business, and then select a color palette that reflects those adjectives.

Stick to a limited color palette

You should use a limited palette of colors for two reasons:

- ✔ From a design perspective, a tight color palette looks better than a rainbow of colors, and it does a better job of showing off photography, product shots, and other elements your Web site showcases. Create a color palette that is comprised of a couple main colors, a couple support colors, a few neutral colors (black, white, off-white, a few shades of gray), and a few accent colors.

- ✔ A limited color palette keeps your Web graphics lean and mean for speedier delivery over the Internet. The fewer colors you use, the smaller the file size. For example, if a file uses just eight colors rather than one hundred, it compresses better and is smaller in file size. The smaller the file size, the faster the file downloads.

Pick colors that work well together

Place swatches of colors next to each other and adjust their saturation, hue, and brightness until you get a set that works well together. Pick colors that complement each other and that you can use for different purposes. For example, pick accent colors that are bright enough for bullets and icons and yet look good on top of your main color.

When choosing a color palette, it's a good idea to include a balance of dark, medium, and light valued colors. If you don't pick any dark colors, your site feels washed out. If you don't pick any light colors, you have nothing to use for backgrounds or setting off special areas.

Use fonts to set the mood

You have to be careful with fonts because they can project a lot of personality. Depending on the font you choose, your Web site can express coolness, professionalism, and everything in between. Just as you choose an appropriate color scheme, you must also choose appropriate fonts for your Web site.

Generally, *serif* fonts (fonts that have little ledges on the tips of each letter, as shown at the top of Figure 7-3) convey feelings of stability, security, professionalism, and longevity — perfect choices for a mutual fund

Figure 7-3: Based on the feel of these fonts, which company would you rather have giving you an MRI?

Web site. The body text of this book, for example, uses a serif font. *Sans-serif* fonts (no ledges on the tips, as shown at the bottom of Figure 7-3) convey feelings of modernism, cleanliness, and agility — great for a site selling advanced technology. Sans-serif fonts are better for short blocks of text and headers. Web text is often set in sans-serif fonts.

Mix fonts wisely

You aren't limited to using just one typeface for your entire Web site. In fact, using a combination of a few fonts for different situations is best: one for headings, another for captions and pullquotes, and yet another for the body text. More than three or four fonts, however, can be excessive. Choose a few different fonts and stick with them. Define a particular font style for each type of element and use it consistently. For example, always use the same font, color, and point size for all headlines, and use another point size and or color for subheads.

Sans-serif heading   Serif text

THE AKITA

The Akita is a large powerful dog that originated in Japan. It was bred for hunting bears in the snow. Today, the breed is a great family companion, although it is not always friendly with the neighbor's dogs.

Figure 7-4: Try using one style for the heading and the other style for the body text.

Serif heading   Sans-serif text

THE AKITA

The Akita is a large powerful dog that originated in Japan. It was bred for hunting bears in the snow. Today, the breed is a great family companion, although it is not always friendly with the neighbor's dogs.

Figure 7-5: In this example, the scheme is reversed.

TIP

Many designers use a mix of both serif and sans-serif fonts for a Web site. You can create a nice look by using one style for a heading and the other style for the body text. For example, Figure 7-4 shows a sans-serif heading and serif body text, and in Figure 7-5, the styles are reversed.

Use graphic elements efficiently

In print design, you can use graphic elements rather freely, but in Web design, large photographs or complex graphics that take up the whole page can be like lumbering elephants on your page for people with slow connections. Big photos mean big file sizes that download slowly. When designing Web pages, try to combine small graphic elements that repeat (background page and table cell tiles), along with small graphics and HTML-generated graphics such as the colored tables and text shown in Figure 7-6. Such a combined approach keeps your page visually rich, yet more efficient.

Figure 7-6: A clever use of background tiles, graphics, and colored HTML elements gives the appearance of a visually rich page that downloads quickly.

Using the ol' grid system

To help you lay out your Web page, set up a grid that you can use to align graphics, text, and HTML elements. A grid can be anything you like — a three-column layout, two horizontal sections, or a page broken into multiple sections, as shown in Figure 7-7.
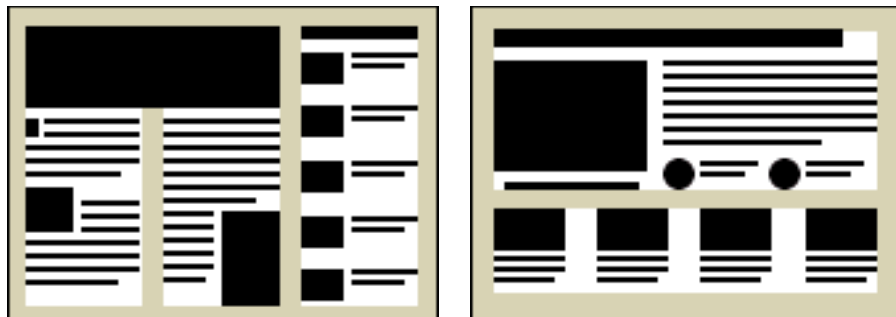
Figure 7-7: These two Web layouts use a grid to divide the page into a few areas.

REMEMBER

Unlike print design, where you can have an angled grid like the one in Figure 7-8, the very nature of HTML, tables, and CSS limit your Web page grid to horizontal and vertical lines.

One way to get around the horizontal and vertical alignment is to place a Flash movie on the Web page that has an angled layout. Unlike HTML, Flash gives you a lot more flexibility in your layouts. The Flash movie is simply embedded in the HTML page just like a graphic, and like a graphic, it ultimately is aligned by your grid system.

Figure 7-8: Angled grids, such as in this print layout, are rare in Web design and are accomplished with a program like Flash.

The grid is unavoidable, and that's a good thing. Grid systems are intended to impose a logical order on your Web page layout. Rather than randomly carving out a spot for everything that is to go on the page, aligning the elements with each other makes the page easier to read, easier to build, and more professional in appearance. Compare the two Web page sketches shown in Figure 7-9. In the example on the left, the elements look thrown on the page. In the example on the right, the same number of elements are neatly presented.
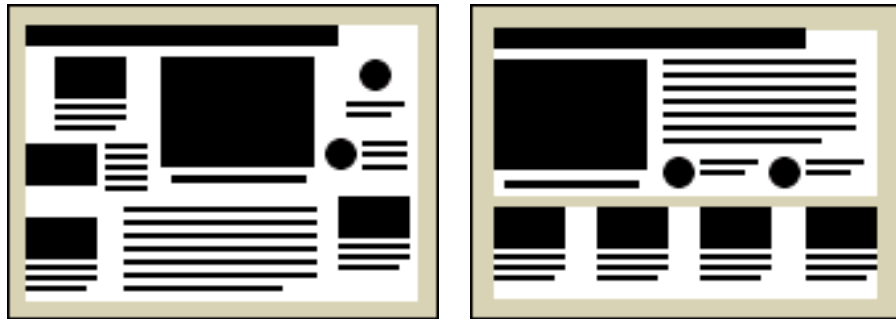
Figure 7-9: A grid system organizes the page and makes it more legible.

Establishing Visual Priority

Clients are often so close to the subject matter that they have a hard time deciding which things are the most important on the page. Their inclination is to make everything big and include as much detail about each item as possible. As the designer, you have to help them create a descending order of importance. If you make everything the same size, everything competes for the user's attention — and nothing stands out.

Compare the two illustrations in Figure 7-10. In the top illustration, it's impossible to determine which section is the main point of the page. Because everything is the same size, no visual priority is established. No one particular element looks more important than another element. And because the layout is so tightly packed together, it looks like it has been subjected to a modern four-horsepower trash compactor. The viewer doesn't get any white space (sometimes referred to as *breathing space*) to take a break from all the information on the page. The bottom illustration has a clear visual hierarchy.

TIP

The key to achieving a balanced, well-prioritized layout is to focus on one thing at a time. Decide which elements are the most important and give them a larger share of the screen space. Also make sure that the important things are within the initial viewable portion of the screen: Don't make

Figure 7-10: Which of these sketches establishes a clear priority for the content?

viewers scroll to reach the most important content. The next few sections offer a few design tips to help you make the most of your screen space without overwhelming your visitor.

Implementing the "big, medium, small" strategy

When designing a Web page, consider it divided into three sections — a big section for the most important stuff, a medium-sized area for the next most important stuff, and the rest of the page for the less important stuff. By thinking of the page in terms of big, medium, and small areas, you automatically limit the amount of detail you can include for the less important things on the page. For example, if the Founder's Story area is one of the least important items on the page, you may leave out a picture and just use a short heading and lead-in sentence that leads to another page with more detail.

Take a look at the home page in Figure 7-11. In this example, the big area has the fruit composition, the medium area on the right has a few specials and a log in. The third most important areas are the top navigation and the search function.

Visually least important

Most important   Second most important
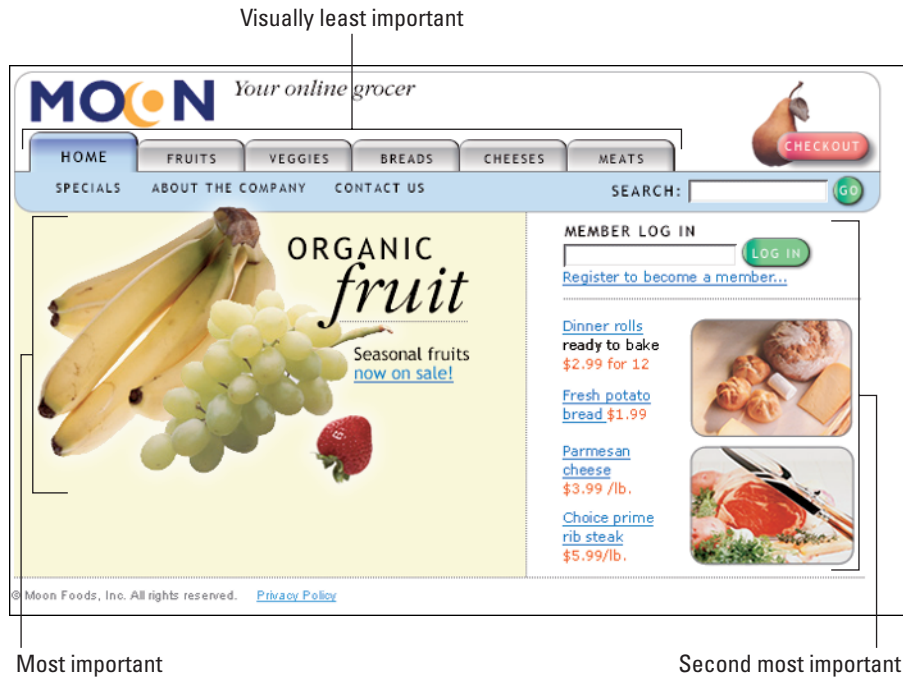
Figure 7-11: This home page is an excellent example of the "big, medium, small" strategy.

TIP

Using a "big, medium, small" strategy prevents you from giving the same visual priority to everything on the page and helps make the page easier to read and navigate.

Breaking up the page into manageable areas

Sometimes you can't avoid the fact that your Web page has a lot of stuff going on. For a news-oriented Web site, for example, breaking the page into a few different areas that you can design separately helps to organize your layout. By using color fields and rule lines to break up the page, you create the illusion of more space because each section operates independently.

For example, the Williams-Sonoma Web site, shown in Figure 7-12, conveys a lot of information, but the three-column layout helps you get through it easily. Notice also how the "big, medium, small" strategy works for graphic images. The big graphic on top cuts across two columns so your eye goes to it first. The smaller graphic on the right is the "medium" area. The content features below is the collective "small" area in terms of visual priority.

Figure 7-12: This Web page conveys a lot of information effectively by using rule lines to break up the space into visually digestible chunks.

Designing around the fold line

If you are a print designer, you're probably familiar with the term *fold line*. The term refers to the spot where a brochure or other kind of printed item folds in half. Anything above the fold line is immediately visible; everything below it is hidden until the viewer unfolds the paper, or performs a similar action. On the Web, the immediately visible portion is pretty small because of the screen size. The point where your Web page gets cut off from view — requiring people to scroll — is called the *fold line*.

Another way that the Web differs from the print world is that the fold line is never quite the same from one computer to the next. Some people have large monitors; some have small ones; and some people purposefully change their default font size settings bigger or smaller. This means you have to design for the worst-case scenario or decide that your site can only support larger monitors. The worst-case scenario is a viewing area of about 800 x 600 pixels, and

when you consider laptop screens and the browser's interface, you have about 20 pixels less in both dimensions. This means that, to accommodate most users, you must place the important stuff within the first 780 x 560 pixels. Otherwise, you run the risk that viewers might not see it, as shown in Figure 7-13.

First some pretty pictures

And now for an important message...

Beware of the evil "fold line"
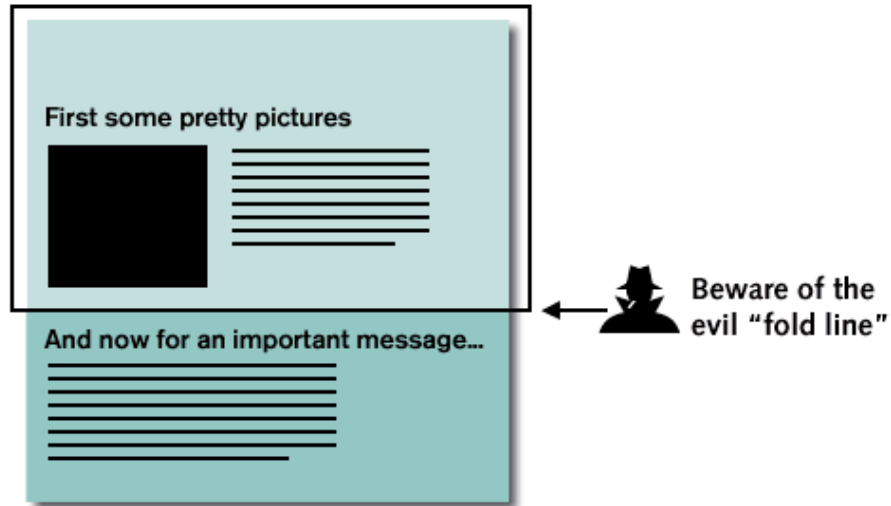
Figure 7-13: Be sure to place important stuff above the *fold line*. Otherwise, viewers may never know it's there.

Adding Breathing Space

Web pages that have tons of stuff packed into every nook and cranny make the page difficult to read and don't give the eye a chance to rest. When building a Web page layout, always plan for some open space around your design elements. The open spaces not only create a more inviting atmosphere that doesn't feel cramped, but also allow the eye to quickly identify all areas of the page — making it more legible.

Here are a few design techniques to open up your layout and create more breathing room for the viewer:

✔ **Use white space wisely.** *White space* is exactly that — large areas of white or light-colored space around your design elements. The physiological effect is an open-air, comfortable feeling. A good goal is to leave at least 25 percent of the page clear of all graphics. Another tip is to leave larger chunks of white space around the most important area of the page,

making it stand out like an island. Apple's Web site at www.apple.com is famous for its extensive use of white space.

Of course, the same principles apply when designing Web sites on a black or dark-colored background, but the feeling that a dark background conveys is somewhat different. A lot of space around objects on a dark background creates a sense of drama and excitement. Figure 7-14 is a good example of using dark space around graphics to create drama.

Figure 7-14: This Web page creates excitement by leaving a lot of dark space around the text and graphics.

✔ **Let some elements float.** Web page layouts are, by nature, fairly geometric and boxy. Although it's normally a good idea to align all your elements to a grid, your page can look too rigid if you follow this rule to the letter. To add visual interest, have some fun with only the most important element on the page. Allow an important element to break the rules a little by falling outside of the grid. This also helps it stand out from the rest. Look at how the design in Figure 7-15 cuts across two columns of the layout.

✔ **Remember that less is more.** Of course, the less you have going on in the page, the easier it is to include white space and floating elements that break out of the rigid grid structure. Limit the amount of detail you include about the less important things on the page — save the detail for another page that focuses on that element.

Figure 7-15: The large image on this page breaks out of the grid to add visual interest to the page.

Staying Consistent

A Web page is a really small window to view a lot of content. Viewing a Web page on a typical monitor is akin to using binoculars to enjoy a breathtaking 180° view — you can see only a small portion at one time. When looking through such a small window, it's easy to lose context and forget where you are in the big picture. That's why it's so important to provide viewers with a consistent user interface that anchors them in your Web site.

Here are three rules for creating a consistent design to keep people oriented in your site:

✔ **Remember — location, location, location!** Always place interactive elements like buttons and links in the same relative location on each page. For example, create a standard navigation bar and find a happy home for it on the page — and leave it there. After you "train" users on how to get around, you don't want to make them search again for navigation.

✔ **Keep the same graphic style.** Always draw interactive elements the same way. If they look like icons on one page and buttons on the next, the visual change throws people off — even if the elements are located consistently. Take a look at the buttons in Figure 7-16. Because the buttons look like icons on Page 2, users may assume they are no longer active links.

Page 1   Page 2
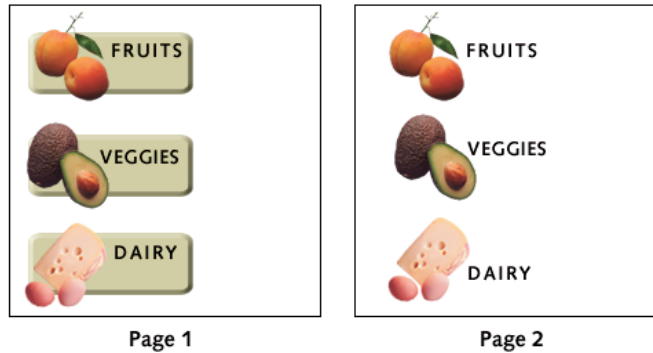
Figure 7-16: The links on Page 1 look like 3-D buttons, but those on Page 2 look like inactive icons.

✔ **Group similar elements on the page.** Not all interface elements are created equal — some links take you to the main sections of the Web site, whereas others take you to resource sections, such as Contact Us areas or policy statements.

Some interactive elements are tools, such as search fields. In your Web page layout, group similar interface elements together on the page and give them a similar graphic treatment. For example, don't mix a Privacy Policy link in with the main navigation set. Instead, set it off in its own area with its own look, along with similar links, as shown in Figure 7-17.

Establishing Design Guidelines

After you have an approved design for the home page and a subpage, you most likely need to clean them up, even the spacing, adjust the navigation, and so on to turn them into production comps. You can use these comps to build a series of production files needed to produce all the needed graphical elements for a site — for example, headlines and buttons — and show how to handle unique layouts — for example, form pages.

Figure 7-17: Give main links a different treatment and separate them from less important links.

Graphic templates

Because Web sites can consist of hundreds of pages, it's not practical or necessary to produce a comp for every page. All that is necessary is to produce a comp that shows how each unique page layout is arranged. For example, all *landing page* layouts are probably the same, so you can do one comp to show how they all look. A landing page is a top-level overview page for one of your primary navigation choices.

Also, to ensure consistency throughout all the graphical elements — from photographs and bullets to interactive buttons — you need to create a series of production templates just for these. For both layout and element production files, use a program like Fireworks or Photoshop and save the files in the software's native format (PNG and PSD respectively).

By saving the production templates in the native formats, you save all the layers, guides, and source art that create the graphics. This way, if the production team needs to change the color or size of a button, or add a new button, as in Figure 7-18, they can work with the source art.
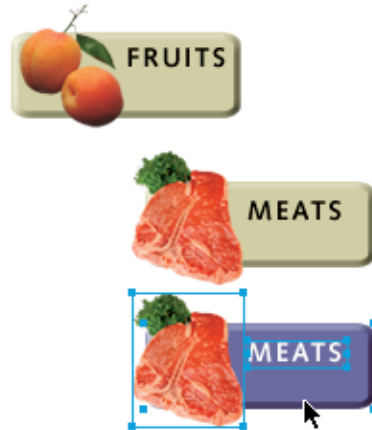
Figure 7-18: When the production team has the source art, making all the buttons for a site is a snap.

Type style guides

To cover all the possible text needs for both graphical and HTML text, you should establish a set of design style guides for the production team. These guidelines can be a simple text document that defines the fonts, point size, colors, and any special formatting and examples. Take a look at the sample set of text style guidelines in Figure 7-19.

Using a program like Macromedia's Dreamweaver, you can easily define a bunch of text styles in a CSS document. CSS, or *Cascading Style Sheets*, is the preferred method of assigning text styles and colors to HTML elements. One CSS document can define all the styles within a site. Each Web page in the site references that one CSS file for style information. For example, if you define a "Headline" style in the source CSS document that is 14 point Verdana bold, with a custom dark red color, any text element in your Web site that is tagged with the "Headline" style is Verdana bold 14 point and red. What's cool, however, is that if you change the red to a dark blue in the source CSS file, all headlines throughout your Web site automagically update accordingly. For more on CSS, see Chapter 14.

Style:	Example:	Font settings:
Text links	About Us About Us	HTML text. Verdana size 1 Link color #333399 Visited link color #990066
Headlines	**Our Company**	Graphic text. Myriad bold condensed, 34 pt, #000066
Subheads	**This quarter's earnings**	HTML text. Arial Black size 2 Color #000066
Body text	This quarter, our company increased revenues by 25 percent.	HTML text. Georgia size 2 Color #333366
Captions	*This graph charts the company's growth over the past year.*	HTML text. Verdana size 1 Color #333366
Break-out text Used for pull quotes, etc.	Our company continues to meet or exceed customer expectations.	Graphic text. Myriad semi-bold Used for pull quotes, etc. Condensed, 16 pt, 125% leading Color #CC3333

Figure 7-19: Example of style guides for both graphical and HTML text for a Web site.

8

Letter-Perfect Type Design

*P*erhaps more than any other detail, your font choices say a lot about the level of your graphic design skills. People who don't have any design training seem to gravitate toward the fun, frilly fonts and — like a kid in a candy store — try out every font on their system. You have to be careful with fonts, however, because they have a powerful effect on the look and feel of your Web site. Of all the graphic elements that go into a Web site, the fonts can make the biggest impression. Fonts have the uncanny ability to make people start listing off adjectives to describe your site like "cheerful" or "serious" or worse, "amateur."

In Web design, you work with two different kinds of text:

➤ *HTML-generated text,* which comprises nearly 90 percent of all text in your Web site, and as such, is the one you work with most often

➤ *Graphic text*, which is actually an image you create in a graphics program such as Fireworks or Photoshop

In this chapter, I discuss basic typography design rules and techniques for creating great-looking HTML and graphic text for the Web. In addition, I discuss text readability issues, what fonts to choose for different purposes, and how to handle browser font display discrepancies. After reading this chapter, you will be able to wield fonts around your Web page with confidence.

Text That You Can Actually Read

Whether a Web site uses graphic or HTML-generated text, reading text on the computer screen is something akin to staring into the sun to watch the airplanes go by: You squint from the glare while trying to figure out the little letters. Reading a lot of text on a Web page is a mild form of torture. You can minimize the torture, however, by limiting the amount of text on each Web page and choosing fonts, font colors, and font sizes that are easy to read on a monitor.

Print designers have the luxury of working with all sorts of fonts and font sizes. This is because they can print with a high *resolution* (fine dot size) on paper — even if the text is very small, readers can still easily see the detail. Images displayed on a computer monitor, however, are a different story. The smallest dot of light on a monitor, called a *pixel*, is $\frac{1}{72}$-inch wide, or 72 dots per linear inch — which, without making you get out your ruler, I can tell you is not that small when compared to standard printing resolutions like 150 and 300 dots per inch.

For example, take a look at the type sample in Figure 8-1. This text is 10-point Syntax in bold on the computer. It has been enlarged so you can see that only a few pixels make up each letter. Such a small number of pixels makes the text appear clunky and unreadable — especially when magnified like this. Ten point type size is a common font size on business cards, so as you can see, what works for a business card doesn't work as well on the computer screen. Of course, some fonts in this size work better than others on the Web, but you get the general idea.
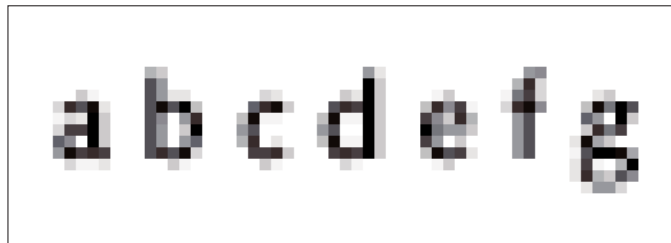
Figure 8-1: A monitor's pixels are too big to render 10 point type effectively.

In addition to fonts and font sizes, other factors contribute to text legibility on a Web page. You can do three things to make your text more readable (regardless of what font you choose):

✔ **Typically, dark text on a light background is more legible than the reverse.** This rule holds true in Web design, so if your Web page has a lot of text that you want people to read, choose a light-colored background with dark text for high contrast.

Also, muted colors are more readable than neon green — or any other bright color. For example, when possible, I like to use dark gray rather than black text for body copy. Dark gray softens the effect, making the text not so glaring. Use bright colors only for headings or short subheadings.

✔ **Try widening the *leading*, or the space between the lines of text, to make the text more readable.** Increasing the leading helps the eye to find the next line of text. Compare the visual effect of the three text blocks in Figure 8-2. The text in the left block is too cramped, whereas the middle block of text is much more readable — especially on the dark background. The leading in the right block gives an entirely different, more decorative feel to the text.

Tight leading   Medium leading   Loose leading

The little boy went down to the river to fetch a pail of water. When he got there, however, he saw that another little boy had beat him to it. Flustered, the little boy went to the other side of the river where the grass was a lot greener.

The little boy went down to the river to fetch a pail of water. When he got there, however, he saw that another little boy had beat him to it. Flustered, the little boy went to the other side of the river where the grass was a lot greener.

The little boy went down to the river to fetch a pail of water. When he got there, however, he saw that another little boy had beat him to it. Flustered, the little boy went to the other side of the river where the grass was a lot greener.

Figure 8-2: Different amounts of leading can lend interesting design effects to your text.

WARNING!

Using a significant amount of leading only works well for smaller blocks of text — not entire pages of body text. You can use a lot of leading to make a lead-in paragraph or a quote stand out from the main body, as shown in Figure 8-3.

The little boy went down to the river to fetch a pail of water. When he got there, however, he saw that another little boy had beat him to it. Flustered, the little boy went to the other side of the river where the grass was a lot greener. The little boy went down to the river to fetch a pail of water. When he got there, however, he saw that another little boy had beat him to it. Flustered, the little boy went to the other side of the river where the grass was a lot greener. The little boy went down to the river to fetch a pail of water. When he got there, however, he saw that another little boy had beat him to it. Flustered, the little boy went to the other side of the river where the grass was a lot greener.

The little boy went down to the river to fetch a pail of water. When he got there, however, he saw that another little boy had beat him to it.

Figure 8-3: Using a lot of leading makes a block of text, such as a caption or a quote, stand out.

✔ **Try to limit each column's width to no more than about five inches.** Because Web pages have more horizontal space than vertical space, you may be tempted to make each line of text run across the entire page to get as much above the fold line as possible. (*Above the fold line* means within the initial browser screen space.) The problem with this tactic, however, is that after your eyes get all the way across the page, you can't easily find the next line. (Plus, tracking your head from side to side like you're eating an ear of corn is a lot of work.) Limiting each column's width to no more than about five inches helps your eyes zigzag down the page quickly and accurately.

Favorite and not-so-favorite fonts for the Web

So many fonts are floating around in the world today that I can't possibly provide you with a complete listing of which ones work for the Web and which ones don't. At the end of this section, however, I do provide a list of my current favorites; but fonts, like fashion, go in and out of style. Beyond my personal list, I can give you some general guidelines to help you select the right fonts for your situation.

For the most part, the rules and suggestions that I list here apply to graphic text that you create to embellish your page. When you use HTML-generated text, your choices are more or less limited to the standard set of fonts available on computers manufactured within the last few years.

Standard versus fancy fonts

In order to choose the right font for the job, you should be familiar with the different categories of fonts. Generally, you should use the standard fonts for pages of body text, and reserve the fancier or decorative fonts for short headings and subheadings, as shown in Figure 8-4.

Decorative headline font

The grass is always greener

The little boy went down to the river to fetch a pail of water. When he got there, however, he saw that another little boy had beat him to it. Flustered, the little boy went to the other side of the river where the grass was a lot greener.

Standard body text font

Figure 8-4: When used together, fancy heading fonts and standard body text fonts create a nice look.

By *standard fonts,* I mean fonts like Garamond, Times, Gill Sans, Helvetica, and Verdana, which were designed for use in text blocks. Font weights for body copy are usually called Roman, Body, Regular. Readers aren't distracted by the design of these classic fonts, enabling them to focus on the text at hand. Standard fonts are designed for easy reading — so they work well in the smaller point sizes.

Fancier fonts, on the other hand, or other weight variants of a font like Bold, Condensed, Semi Bold, Black, and Ultra, have a lot of detail and personality — ideal for larger-set headings and subheadings. These fonts grab people's attention, but they aren't designed for a long page of text, as Figure 8-5 demonstrates. You can often distinguish the decorative fonts just by their names, such as Kid Print, Comic Sans, and so on.

Easy to read font for body text   Decorative font for headings

The little boy went down to the river to fetch a pail of water. When he got there, however, he saw that another little boy had beat him to it. Flustered, the little boy went to the other side of the river where the grass was a lot greener.

The little boy went down to the river to fetch a pail of water. When he got there, however, he saw that another little boy had beat him to it. Flustered, the little boy went to the other side of the river where the grass was a lot greener.

Figure 8-5: The font on the left is designed for legibility in large blocks of text; the font on the right, however, should be reserved for headings.

Through the thick and thin

WARNING!

The thickness of each individual letter (or *line weight*) contributes greatly to a font's readability. Because pixels on computers aren't comparatively very small, fonts that are too thin can fade away into oblivion — especially at smaller point sizes.

If you examine any font, you notice that the thickness varies in each letter. For example, take a look at the letter *m* in Figure 8-6. The letter is set in a few different typefaces for comparison. Some fonts, such as Kepel Bold (the example on the far left), change drastically between the thick and thin areas. At a small point size, all you see is a blob of bold strokes — the thin strokes disappear. When choosing fonts for small point size text, therefore, find versions that don't have a big variance in the stroke thickness — like the middle and right examples in Figure 8-6. Also, avoid the "light" weight editions of fonts — they are too thin to read at a small point size. (Most font families come in a variety of thicknesses, ranging from light for the thinnest of the set to bold, black, and ultra for a progressively heavier line weight.)

TIP

Figures 8-7 and 8-8 list some serif and sans-serif fonts that read well on the Web. The fonts on the list all survived the "10 point test": They are all readable when set at just 10 points, which is the smallest font size for text that you should ever consider in Web design.

These two figures show fonts that pass the 10-point test ideal for graphic or HTML text. For graphic text, you can use whatever font works best for you. For HTML text, you should choose only fonts that are commonly available on most computers. For a list of these, flip ahead to Figure 8-18.

12 point

m m m

m m m

12 point zoomed in

This font is difficult to read when small.
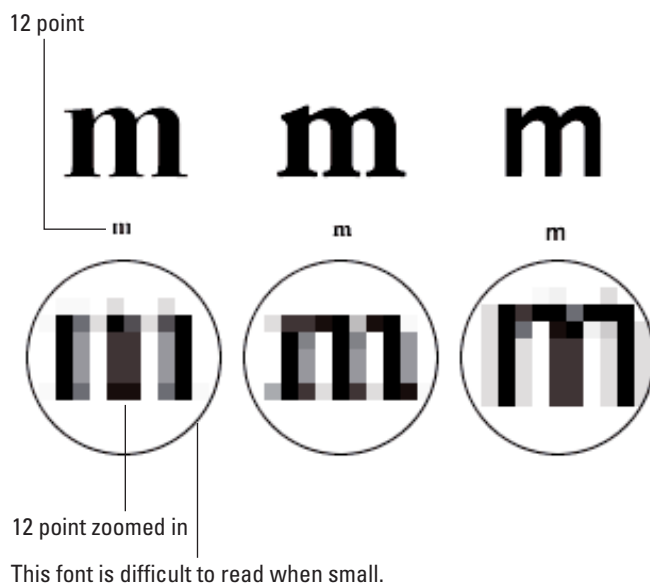
Figure 8-6: The font on the left differs dramatically between the thick and thin areas.

Font:	Example:	Comments:
Arial and Helvetica	Favorite Sans-Serif **Favorite Sans-Serif** Favorite Sans-Serif **Favorite Sans-Serif**	Works well for body text and headlines though can look a little generic
Meta	Favorite Sans-Serif **Favorite Sans-Serif**	Similar to Officina, works well for all situations
Myriad	Favorite Sans-Serif **Favorite Sans-Serif**	Clean modern font for all situations
Officina	Favorite Sans-Serif **Favorite Sans-Serif**	Modern san serif that works well for body text and headlines
Syntax	Favorite Sans-Serif **Favorite Sans-Serif**	Clean modern font for all situations
Verdana	Favorite Sans-Serif **Favorite Sans-Serif**	Designed specifically for the Web, Verdana works especially well as HTML text.

Figure 8-7: Sans-serif fonts for graphic or HTML text that pass the "10 point test."

Font:	Example:	Comments:
Bookman	**Favorite Serif**	Bookman is a more blocky serif, works well for headlines. Bold works best for the smaller font sizes
Century Schoolbook	**Favorite Serif**	The bold variety is readable but wider than other fonts the same point size
Copperplate	**FAVORITE SERIF**	Copperplate is usually all capital letters. Its formal nature works well for business headlines
Garamond and Goudy	**Favorite Serif** **Favorite Serif**	Beautiful old-style fonts for classic looks. Good for headlines or body text. Bold works better for small point sizes
Georgia	Favorite Serif	Like Verdana, Georgia was also specifically designed for Web legibility. The font works well for body text and headlines.
Times	**Favorite Serif**	To me, this font has less character than the other fonts listed here, but is a good stand-by for body and headline text.

Figure 8-8: Serif fonts that pass the "10 point test."

Serif versus sans-serif

Your font education won't be complete until you know the difference between serif and sans-serif fonts. As Figure 8-9 illustrates, a *serif* is the little ledge that adorns the tips of a letter. *Sans-serif* fonts, on the other hand, don't have these ledges. Some modern sans-serif fonts do have a little curl on their tips, making you look twice, but they're still considered sans-serif fonts.

Ledge   No ledge   Close, but no cigar

Figure 8-9: The letters of serif fonts have ledges on their tips. Sans-serif fonts do not.

REMEMBER

You can use either a serif or a sans-serif font for both your body text and headings. Both styles work just as well for both purposes. The difference really comes down to the unique look you get from each type. Generally, a serif font conveys a more formal, business-like feel, whereas a sans-serif font elicits a clean, modern look.

TIP

Try mixing serif and sans-serif fonts together. For example, use one style for headings and the other style for the body text. Choosing a set of two to four fonts for your Web site — one for each different kind of type element — helps to add visual interest to the page.

Not too big; not too small

Unless you're going for a postmodern, unreadable artistic effect, avoid graphic text that is less than 10 points in size. Some fonts do work fairly well at 10 points, but most don't. On the flip side, don't make your text so big and horsey that your Web page looks like an eye chart. (By the way, *horsey* is a great, fussy designer term for large and clunky.)

A good rule is to keep your body text somewhere in the range of 10 to 12 points (some fonts are naturally larger than other fonts). Fourteen-point body text is way too big. For headings, use 14 to 24 point sizes. You have to use your judgment for each situation, but headings larger than 24 points approach the horsey territory.

TECHNICAL STUFF

Always use even-numbered point sizes like 10, 12, 14, 16, and so on — especially for sizes in this range. Computer fonts are *hinted* (crafted) to look good on the screen in these specific point sizes, generally even numbered sizes. If you specify an odd size like 11 point, the computer must scale the font, which tweaks its appearance.

Text on background tiles

Many designers use background tiles to spice up the design of their Web page. A background tile is a repeating pattern that fills the entire Web page. The design elements are placed directly on top of the tiled pattern, just as they are on a solid-colored background. To make a background tile, create a graphic of any size. The Web page then repeats this graphic end-to-end across the entire page.

The problem with background tiles, however, is that their busy patterns and competing colors can adversely affect the visibility of your text. You can use background tiles; you just need to prepare them properly.

Here are some design techniques for creating background tiles that work well with text:

- ✔ **Create a background tile that is much larger than your Web page.** Many people think of background tiles as a small, square 1-x-1-inch graphic like the floor tile in a house. This isn't necessarily true. You can create a background tile that is much larger than your Web page! This way, the single tile extends past the browser window, so users don't see it repeat unless they do a lot of scrolling. By creating a large tile, you can concentrate busier designs away from the main text areas.

- ✔ **Create a background tile that is a long and skinny horizontal strip.** The pattern repeats *down* the page, but users don't see it repeat *across* the page unless they scroll, as shown in Figure 8-10. The net effect of this technique is the same as the "whole page" tile technique (discussed in the previous paragraph): You can control the placement of the busy and not-so-busy areas so they don't interfere with your text. If you use a large tile, keep the design simple and use as few colors as possible.

- ✔ **If you use a smaller tile, make sure that the pattern is subtle.** If you use a small tile with a pattern that repeats across the page, be sure that the colors you choose don't overpower the colors of the other text and graphics on the page. For example, make a tile with a subtle pattern of a few light colors and then choose a dark font for as much contrast as possible.

A long thin tile repeats down the page.
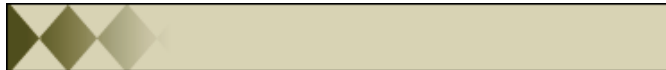
The Web page filled with this tile.
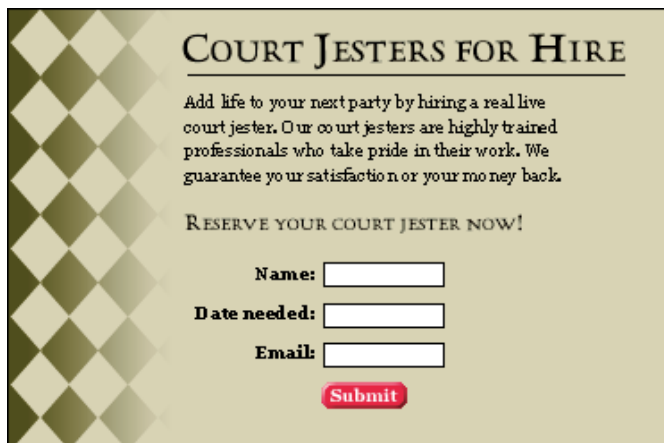
Figure 8-10: You can control the placement of background textures by creating a long, narrow tile that repeats down the page.

Graphic text versus HTML-generated text

Web pages contain two types of text: graphic text (which is really an image) and HTML text (which is generated with HTML tags). HTML-generated text is considered *live* because it is fully editable — just like the text in your word processor. You can apply various font and size settings to HTML text, and you can update it easily. Graphic text, however, is just an image. To make corrections to the text, you must re-make the graphic. In addition, graphic text has a larger file size than its equivalent in HTML-generated text, so it can take longer to download.

For these reasons, using more HTML text than graphic text on your Web site is best. Because HTML-generated text is much more efficient, you may be wondering why you should use graphic text at all. Together, these two types of text offer different levels of design flexibility. Because graphic text is an image just like a photograph, the possibilities are endless in terms of font, color, and even texture choices (such as placing an image of a sunset inside the letters).

Graphic text also allows you to achieve soft *anti-aliased* edges around the letters in your text, as shown in the zoomed-in detail of Figure 8-11. Because pixels are square, the rounded curves of graphics and graphic text look like stair steps. Anti-aliasing places a small gradient of color to softly blend the curved edges into the background. When you use a graphics program like Photoshop or Fireworks to create graphic text elements, the programs anti-alias the text by default.

HTML text, on the other hand, has *aliased* edges, so you get the stair-step effect, as shown in Figure 8-12. Also, with HTML-generated text, your choices are much more limited because the fonts that you specify must be available on the end user's system. With so many different systems in use, you have about four fonts to choose from. In the HTML tags, you can specify first, second, and third choices for a font. For example, if you want a header to be Verdana, you can specify Arial and Helvetica as alternate font choices in case an end user doesn't have Verdana. I discuss this further later in this section.
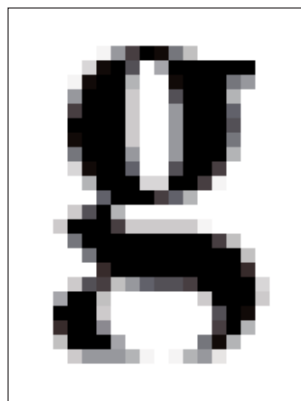
Figure 8-11: Anti-aliasing places a tiny gradient around curves to softly blend them into the background color.
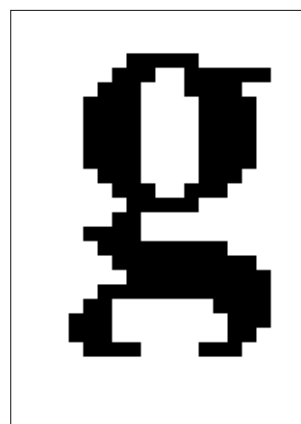
Figure 8-12: Aliased graphics have no blending on their edges — making the curved areas look like steps.

A dash of graphic text, a pound of HTML text

Most Web sites need to be easily updatable. Making changes to HTML-generated text is a relatively easy process, but re-doing a whole slew of graphics is a grind. For this reason, you should use a ratio of about 90 percent HTML text to 10 percent graphic text. Only use graphic text for headings, buttons, quotes, and other decorative text elements that you can sprinkle throughout your layout.

TIP

If you do create graphic text, you should always save the source art in the native format of the software. For example, if you create a heading in Photoshop and export it as a GIF for the Web, you should also save the original Photoshop file. After you export a file as a GIF, you can't change it — you have to re-make it. Software programs such as Photoshop and Fireworks save text in an editable format so if you ever need to make changes to the GIF, you simply go back to the source file, re-type the changes, and then re-export as a GIF or JPEG. For more information on GIF and other file formats, see Chapter 12.

WARNING!

You should also keep graphic text to a minimum if you plan to *localize* (lingo for *translate*) your Web site into other languages. Translating HTML-generated text is far easier than rounding up all the sources of your graphic text and then re-making them in Spanish versions.

Graphic headings

After you get a good handle on where to use graphic text in your Web page layout, the next step is to dig in and create the graphic text. Graphic text is a cinch to make: Simply launch your favorite graphics program, wield the Text tool around a little, and add any effects as needed.

In this example, you use Fireworks to create a heading for a Web page, complete with drop shadow. The steps are pretty much the same if you use Photoshop:

1. **Launch Fireworks.**

   Fireworks appears, as shown in Figure 8-13. Note the location of the Tools panel, Property Inspector, and other panels.

2. **Choose File⇨New from the menu bar.**

   The New Document dialog box appears, as shown in Figure 8-14. You can also create a new file by pressing ⌘+N (Mac) or Ctrl+N (Windows).

3. **Specify the dimension, resolution, and canvas color for the new file, and then click OK.**

Tools panel

Panels

Property Inspector
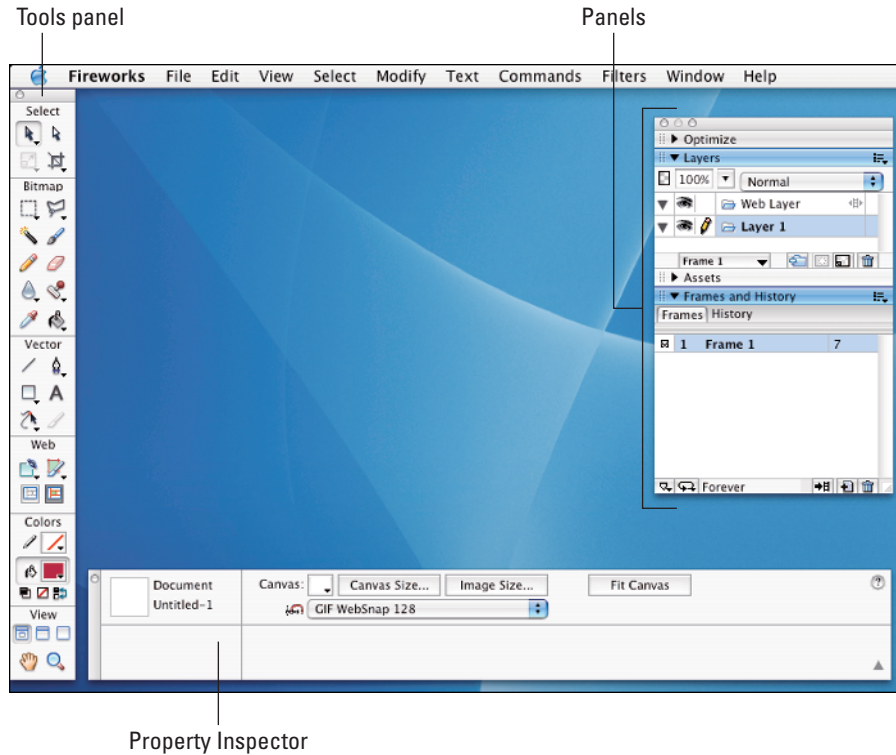
Figure 8-13: The Fireworks interface.
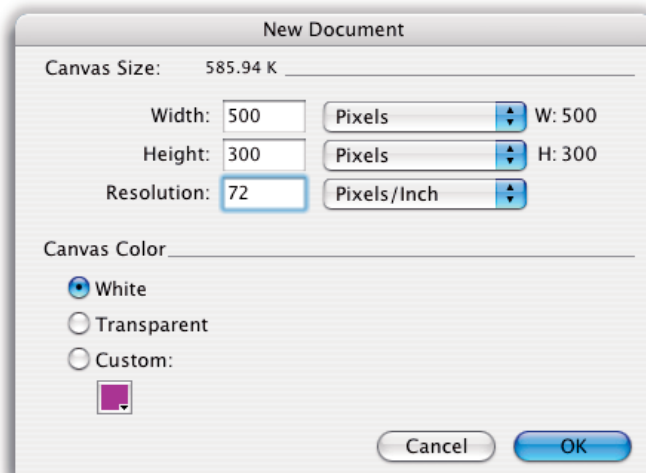
Figure 8-14: Use the New Document dialog box to specify dimension, resolution, and background canvas color for your new file.

To give yourself enough room to work, enter 500 pixels for the width and 300 pixels for the height in the Canvas Size area, and choose 72 dpi. If you export without trimming it further, the graphic will be 500 x 300 pixels in size. This size is too large for a header, but I like to start with a big canvas. Later in these steps, you will crop the canvas to match the size of the header. Cropping the canvas to the size of your graphics is always best. Smaller dimensions result in a smaller file size, which in turn speeds up download performance.

Please note that while the figures shown in these steps show the Macintosh interface, Fireworks works the same on both Mac and Windows. Windows users can follow the steps equally well.

For this document, I want to build a heading for a Web page with a white background (the color of a Web page is set in the HTML code for the page), so I leave the Canvas Color (or background color) white. When building graphics, you should always use a background color that matches the color of your Web page background.

When you build graphics in Fireworks, you work in Original mode (note the tabs at the top of the document shown in Figure 8-15). Later in these steps, you'll use the Preview mode to see how your palette and file format choices affect the quality of your design.

Original mode
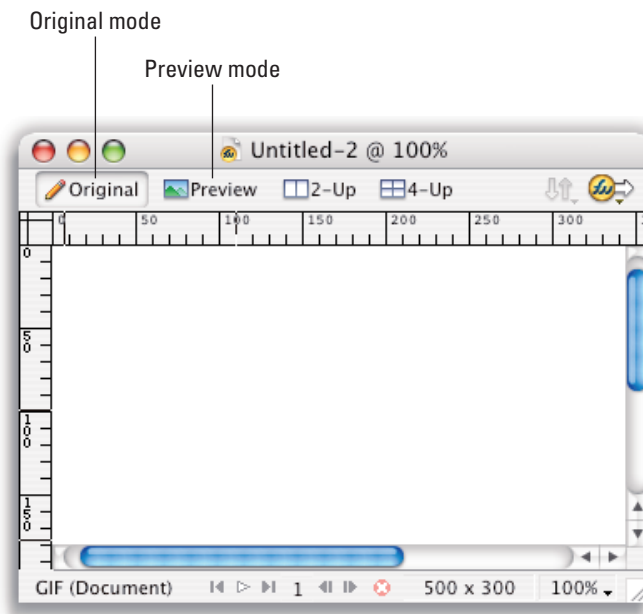
Preview mode

Figure 8-15: Build graphics in Original mode. Click the Preview tab to see how it will look when exported. Click the Original tab to continue editing.

4. **Select the Text tool from the Tools panel (it looks like the letter A), and click anywhere in your document to begin typing. When finished, click the black pointer tool in the Tools panel (upper-left) to exit typing mode.**

5. **Apply font, size, and color settings to your text.**

   With the pointer tool, click your text object to select it. In the Property Inspector (the long horizontal panel shown in Figure 8-16), select a font from the menu, and then enter point size (to the right of the font choices). Click the color swatch to the right of the point size and select a color from the pop-up palette.
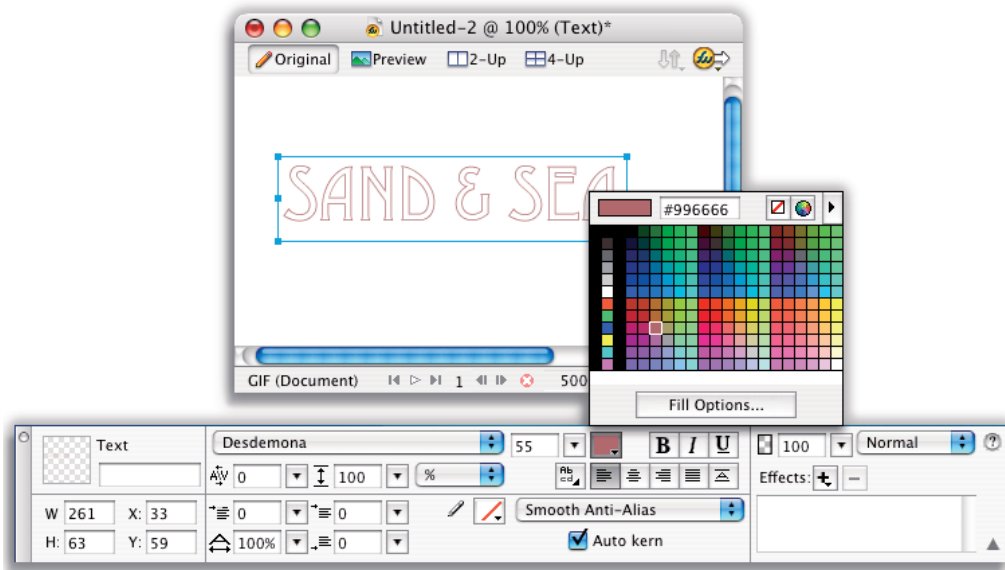
Figure 8-16: Use the Property Inspector to apply color, size, and font choices to your text.

6. **Add effects to your text.**

   While the text object is still selected, follow these steps:

   1. Choose an effect for your text by clicking the + symbol on the right side of the Property Inspector and selecting from the list of choices.

   2. After you select something (such as Shadow and Glow⇨Drop Shadow), a small dialog box appears where you can adjust settings. Play with the settings until you like the result. You'll notice that as you play, the text on-screen dynamically updates.

   3. To keep your settings, simply click anywhere outside the dialog box.

4. If you ever want to update your drop shadow settings, click the i icon next to the Drop Shadow effect. You can add multiple effects. Just click the + icon and choose another effect.

5. To remove effects, highlight it in the list and click the – icon.

7. Trim the canvas to just the text.

When you finish with the design, you can prepare it for Web export. If you don't trim the canvas, all the surrounding space gets included when you export the file, making the graphic a big white block with a piece of text swimming in the center. To trim the canvas size, choose Modify⇨Canvas⇨Trim Canvas from the menu.

8. Choose a file format.

Before exporting this heading, apply a Web file format setting. In the Optimize panel, shown in Figure 8-17, choose the following options:

1. Choose GIF from the first list and Adaptive from the second list, as shown in the figure.

2. Next to Colors, choose 128. This limits your heading to no more than 128 colors. To optimize the image further, you can enter less colors. The fewer colors you use, the smaller the file size is and thus, the faster it downloads on the Web. Choosing too few colors, however, can degrade the quality of the image.

In Fireworks, you can preview how your color palette and file type settings will look by clicking the Preview tab at the top of the document window. While in Preview mode, you can try different Web file settings in the Optimize panel to see how each look and how long each will take to download (the Preview mode tells you). When you have a setting that you like, click the Original tab.

Select Adaptive palette

Select GIF format
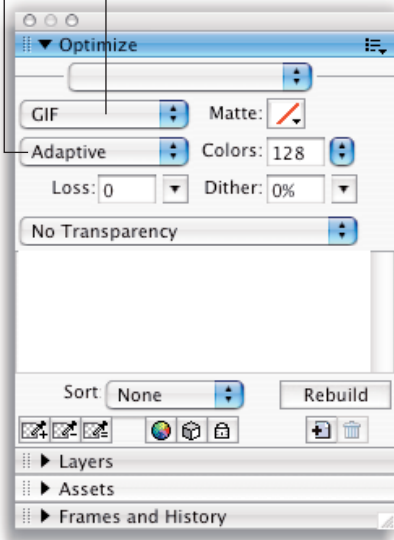
Figure 8-17: Use the Optimize panel to specify Web export settings for your graphic.

9. Save the editable file and export the heading.

That's it! Before exporting for the Web, save your work in the native Fireworks format by choosing File⇨Save from the menu bar. In the Save dialog box, enter a name and keep the `.png` extension after the name

(such as `myfile.png`). Save it wherever you like on your hard drive. By saving the original PNG file, you can always reopen it, make changes to the design, and then export it once again as a GIF or a JPEG for the Web.

To export this heading to the Web, choose File➪Export. In the Export dialog box that appears, select Images Only (next to Save As). Enter a name for your file (keep the `.gif` or `.jpg` at the end), select a folder on your hard drive, and click Save.

Controlling Text Display

Because most of your page is HTML text, you should spend some time going over the lay of the browser font land. When you place text into a Web page, you have limited control over its formatting. You can use HTML tags or CSS (Cascading Style Sheets) to apply formatting, such as color, bold, italic, underline, and font size settings. To some degree, you can also specify a font.

When you specify a font in HTML or CSS, you are relying on the user's computer to have that font and use it to display your page correctly. If a computer doesn't have the font that the page calls for, the browser substitutes a font. This can wreak havoc on your page layout, because as you may know, fonts vary a great deal in size and width — even when set to the same point size.

HTML pages load from top to bottom in a linear flow. For this reason, if the substituted font is larger than the one you specify, the layout shifts downward significantly.

Font specifications

Use fonts that practically all users have on their computers, such as Helvetica, Arial, Times, Verdana, Georgia, and so on. Take a look at Figure 8-18 for a listing of the standard browser fonts — it's not a long list.

Another safeguard that you can take is to specify a *preference* list of fonts in your HTML tag. For example, when you write the font tag, list the preferred font first, and then follow it with an alternative list of fonts to use if the first font isn't available. When someone loads the page, the browser scans the list until it finds a match — ideally finding the first font in your list. Here's what the HTML tag looks like:

```
<font face="Verdana, Arial, Helvetica, sans-serif"> Make a
             preference list of fonts </font>
```

In this case, Verdana is the first choice, followed by Arial and Helvetica. The last entry, sans-serif, tells the browser to insert any available sans-serif font on the system if it doesn't have Verdana, Arial, or Helvetica.

Font:	Example:	Size 2 Text Sample
Verdana	Sample Text	The quick brown fox tripped over the stump and slid into dog's den.
Arial	Sample Text	The quick brown fox tripped over the stump and slid into dog's den.
Arial Narrow	Sample Text	The quick brown fox tripped over the stump and slid into dog's den.
Helvetica	Sample Text	The quick brown fox tripped over the stump and slid into dog's den.
Georgia	Sample Text	The quick brown fox tripped over the stump and slid into dog's den.
Times New Roman	Sample Text	The quick brown fox tripped over the stump and slid into dog's den.
Times	Sample Text	The quick brown fox tripped over the stump and slid into dog's den.
Courier	Sample Text	The quick brown fox tripped over the stump and slid into dog's den.
Courier New	Sample Text	The quick brown fox tripped over the stump and slid into dog's den.

Figure 8-18: Standard browser fonts available on both PCs and Macs.

CSS Font Control

Cascading Style Sheets (CSS) technology allows you to set up your own special set of font styles, which you can use consistently throughout your Web site. For example, imagine that you are designing a fairly large Web site and you want to ensure a consistent look across each page. All headings should be Arial Narrow Bold, 18-point (yes, you can use real point sizes in CSS!), and a dark red color. All captions should be Verdana Italic, 10-point, and steel blue. Note that it's still important to specify standard browser fonts (refer to Figure 8-18). The main advantage of CSS technology is that it saves you time by allowing you to quickly apply consistent settings to all text elements throughout a Web site.

CSS comes in two flavors: *external* style sheets and *internal* style sheets, as described in the following two sections.

External style sheets

For external style sheets, you define a bunch of text styles in one CCS file like `mystyles.css`. Each style in the file can have a name like Headline, Caption, and so on. For each style, you assign a font, color, and size. Then, each of your Web pages reference this separate `mystyles.css` file for their font information. Text elements on each Web page reference one of the CSS styles in the CSS file. This way, if you ever have the need to make a global font change, like change the Headline style from dark red to blue, you make the change once in the CSS file, and all text elements throughout the Web site that use the Headline style update instantly.

Internal style sheets

Internal style sheets are styles that you define for only one page — not the whole Web site. Internal style sheets are convenient for small nuances that are only found on one page, as they are found inside the `<head>` tag and override external style sheets. The style information and the style name go in the top `<head>` section of the HTML code, as shown in this example:

```
<style type="text/css">
.headline {  font-family: "Arial Narrow Bold"; font-size:
            18pt; color: #990033}
.caption {  font-family: Verdana, Arial, Helvetica, sans-
            serif; font-size: 10pt; font-style: italic; color:
            #333366}
</style>
```

In this code syntax, you can see that the `<heading>` style is Arial Narrow Bold in 18-point. The funny-looking number after `color` is a *hexadecimal* code that creates a dark red color. As you add headings to your Web page, you can quickly format them by applying the `<heading>` style that you defined for the page:

```
<span class="headline">Headlines are a cinch to format</span>
```

Style sheets are cool because they make it remarkably easy to update the look of your page. For example, imagine that your clients change their branding and now want a whole new set of font choices and colors. Yikes! This would normally require tedious hours of selecting and updating each individual text element on the page. If you used CSS, however, you simply need to update the style sheet definitions in the Head section, and — voilà — the whole page updates.

9

Color on the Web

*T*he way that color works on the computer is one of those great mysteries that everyone appreciates but has no idea how it works — including most Web designers. Most people figure that it's just too complex to understand. And designers don't care about the inner workings — they just want their stuff to look good. But in the print world, designers must know how the color printing process works in order to get the most out of their designs. The same rule holds true in Web design — you simply can't get around the technical stuff. You have to know how color works on the Web in order to design effective Web pages.

Never fear, the technical mysteries aren't what they're cracked up to be. Whereas the printing process uses four colors (*c*yan, *m*agenta, *y*ellow, and blac*k,* or the CMYK system) to mix a wide range of colors, the computer uses only three colors (red, green, and blue) to generate its colors. What's amazing is that even though the computer uses one less color than its print counterpart in its mixing scheme, those three colors are capable of producing a mind-blowing range of hues — way beyond the possibilities of the CMYK system. There's a catch, of course: Older systems can display only a limited subset of these colors with any reliability. This limited subset of colors is called the *Web-safe* palette. This means that you have to be selective in your color

#CC3333

Fill Options...
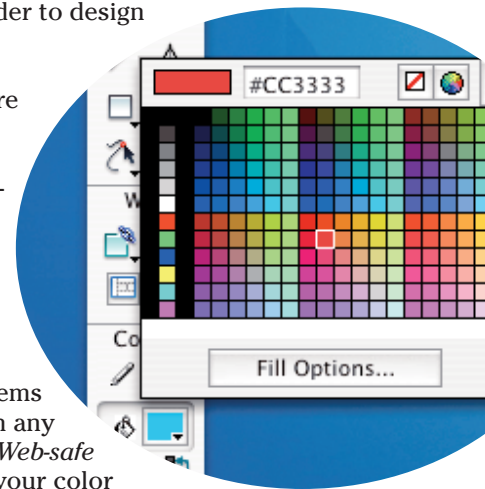
choices for Web graphics, or you may have unexpected results. The way you use color in your graphics also has a great impact on their download efficiency for the Web. For example, using fewer colors results in smaller file sizes, which speeds up delivery over the Internet.

If you read this chapter in its entirety, you'll understand more than you ever wanted to know about the mechanics of color on the computer and how to use color palettes properly in your Web graphics.

The Secret World of RGB

The secret to all the colors you see on your computer monitor is a system called *RGB color*. This system uses just three colors (red, green, and blue) combined together to create all colors (including black and white) on your monitor. Although this sounds like quite a feat, consider that the system works in a way similar to natural daylight. When the sun is up, you see white light. When the sun goes away, it's pitch black outside. Only if you use a prism to break up the spectrum can you see all the colors contained within white light. Like the sun, RGB color is an *additive* color process. The CMYK printing process, on the other hand, is a *subtractive* color process. I explain the difference in the following section.

Subtractive and additive colors

When you mix two or more colors together to create a new color in any medium — whether it's in print or on a computer screen — the method you use is either an *additive* or a *subtractive* process. These terms refer to the way you produce the color white.

In printing, you achieve the color white by mixing no colors at all. By removing (subtracting) all colors, you leave the color of the plain paper — which is presumably white — shining through. Therefore, the print process is a subtractive color process. The only way to print white is by printing nothing at all — subtracting all colors from the page.

Combining cyan, magenta, yellow, and black inks in various proportions creates all the other colors. The color we perceive on the page is the result of the different spectral colors that the inks absorb.

At the other end of the spectrum (pardon the pun) is the computer's *additive* process. To create white on a computer screen, the display mixes all three colors — red, green, and blue — together at full strength. Just as if you switched on a light bulb, all colors of the spectrum come together to make white. And just as if you turned off a light switch, removing all the RGB colors leaves you with black.

Gazillions of colors

How do you mix all the colors of the rainbow using this *additive* system of just three colors? As with the CMYK process, the answer is to use different amounts of the three colors. The RGB color scheme contains 256 *levels* (variations numbered from 0 to 255) of red, 256 levels of green, and 256 levels of blue.

To generate (or mix) a color, you take one level of red and mix it with one level of green and one level of blue. For example, to make a nice turquoise color, you can mix red number 75, green number 199, and blue number 211, as shown in Figure 9-1. In this scenario, red — set at just number 75 — is used at the weakest level. This makes sense because turquoise is primarily a mix of green and blue shades.
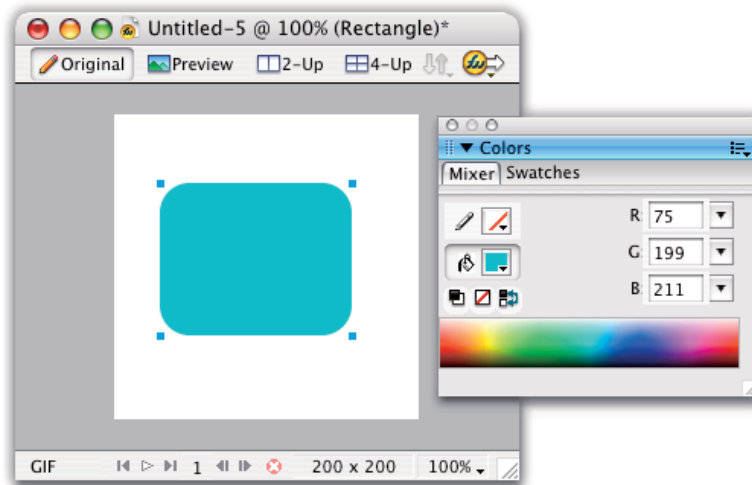
Figure 9-1: To make a turquoise color, mix red, green, and blue light at various levels.

If each color (red, green, and blue) has 256 possible settings, how many colors can you mix with the RBG system? For those of you who are mathematically inclined, you know that the number of colors is 256 x 256 x 256, or 56^3, as shown in Figure 9-2. This number amounts to a whopping 16,777,216 different color combinations — many more than you and I can see with our eyes!
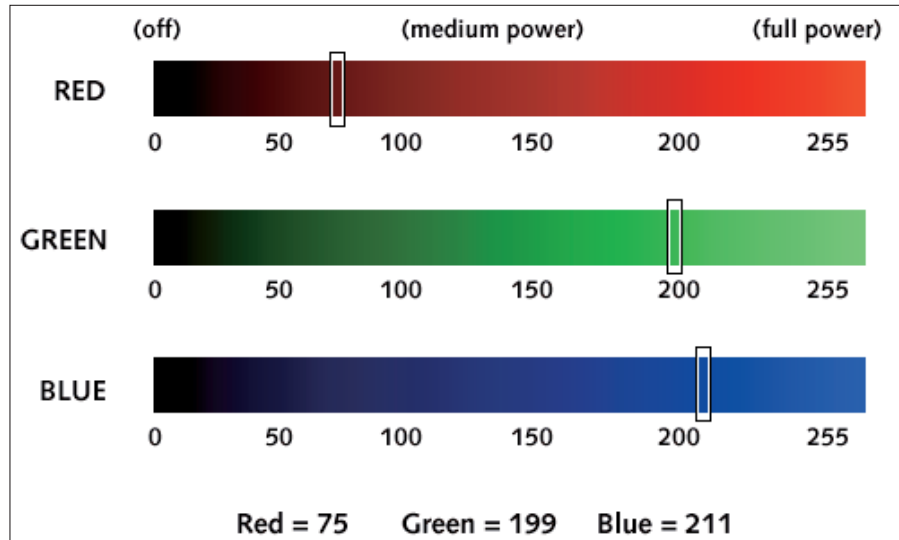
Figure 9-2: You can create more than 16 million colors by mixing all possible levels of red, green, and blue.

Color bit depth

Although the computer's RGB system is capable of mixing 16 million different colors, the monitor is not always capable of *displaying* all these colors. To help you better understand this concept, I explain *bit depth* and how it relates to Web design in the following paragraphs.

A *bit* is a tiny segment of computer information — it's like the atom of computer data. Without getting too technical, a lot of bits are needed to make colors appear in an image. The more colors that an image contains, the more bits of data that it has, and thus the higher its *bit depth*. For example, a 1-bit image has just two colors. An 8-bit image can have up to 256 colors.

The monitor has to work hard to display all these colors contained in an image. Some monitors simply don't have enough computer memory to display all the color data. Therefore, you may encounter a situation where the image has more colors than a monitor can actually show.

Terms like *24-bit color* refer to both images and computer monitors. A 24-bit monitor can display 24 bits of color information. A 24-bit image has a lot of colors — specifically, the millions that are possible with the RGB system. Here's more detail:

- ✔ **24-bit monitor.** Although the RGB system can make 16 million different colors, a computer monitor needs a lot of power (a.k.a *VRAM*) to show all these colors. More specifically, one monitor pixel uses 8 bits of data to display one of 256 colors. To display the full spectrum of all three colors, each pixel needs 24 bits. Such 24-bit monitors, often called *true color* monitors, give gorgeous continuous tone results. Most modern monitors are true color monitors. If your Web audience uses older computers, color bit depth is something you need to be mindful of.

- ✔ **24-bit image.** A 24-bit image is one that is capable of displaying all 16 million colors of the RBG color space. This is not to say that every image *needs* all 16 million colors — that would be quite the psychedelic image. More accurately, a 24-bit image has unlimited *access* to all the colors in the world as needed to create a beautiful continuous tone. Color wise, using a 24-bit image is like having a blank check in Monte Carlo.

The most colorful image in the world, however, won't look good if the monitor on which it's displayed can't support it. It is possible to have a nice, millions-of-colors, 24-bit image that you can't display properly because your monitor only supports 8-bit or 16-bit images. What happens, then, to the image? On an 8- or 16-bit monitor, a 24-bit image is simplified into what looks like a work of pointillism art.

Figure 9-3 shows the same 24-bit image as seen on two different monitors. On the left, the image is displayed on a 24-bit monitor, so it looks great even when you zoom in. The image on the right shows what the image looks like on an 8-bit monitor — very grainy. Because an 8-bit monitor can only display 256 colors at one time, it uses a pointillism-like approach called *dithering* to approximate the millions of colors in the image. Dithering places various colored dots near each other so that from a distance, they appear to create one color. For example, dithering places a yellow dot and a red dot close together so that from far away, the eye sees orange.

But what does an 8-bit image look like when displayed on a 24-bit monitor? When the image is of low quality to begin with, viewing it on a high-quality monitor doesn't improve anything. If an image has been reduced to an 8-bit palette of just 256 colors, the monitor can do nothing to resurrect the lost millions of colors. The image looks dithered — regardless of the monitor's bit depth.

Image viewed on a
24-bit monitor

Same image appears
dithered on an 8-bit monitor
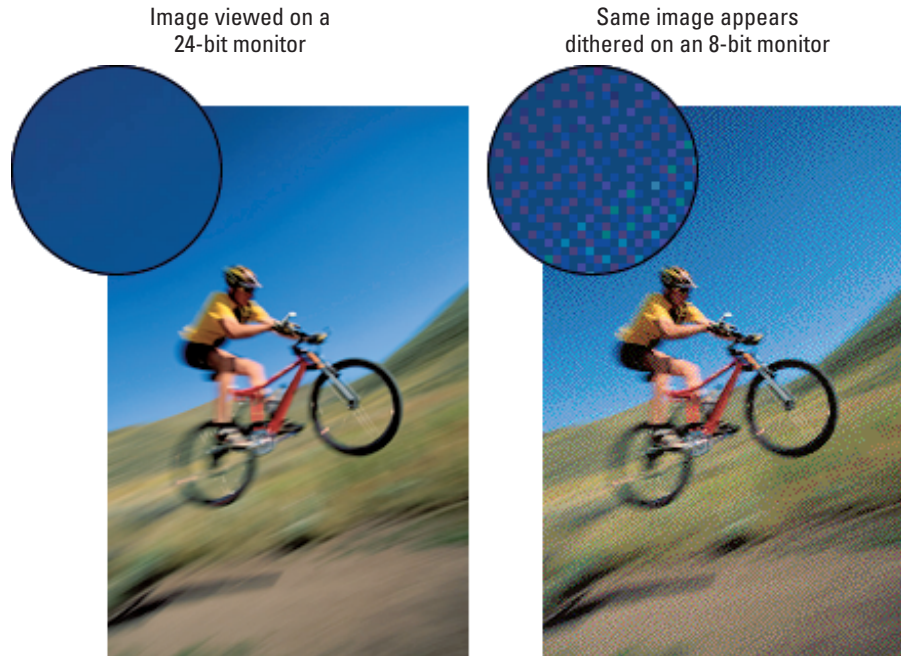
Figure 9-3: The images in this example are both 24 bit. One looks grainy, however, because it's displayed on an 8-bit monitor.

As I discuss earlier in this chapter, a computer uses 8 bits of data to show 256 colors. Each of the red, green, and blue primary colors has 256 levels. So does this mean that an 8-bit image is monotone, showing all 256 possible levels of red? The answer is no — an 8-bit image simply uses a select palette of 256 colors. These colors are a subset of all 16,777,216 possible colors. The decision that you as a designer must make is *what* subset of colors you want to use. To explain this concept further, I discuss palettes and how they work on the Web in the following section.

The Web-Safe Color Palette

The best reason to reduce the number of colors in an image — from millions down to just 256 — is to reduce the file's size. Small file size means faster transmission. You must sacrifice quality and detail for the sake of optimal delivery over the Internet. You can choose from among a few pre-fab palettes, or you can make your own. ***Remember:*** The fewer colors you choose for your palette, the smaller the file size.

To add to the complexity, only certain colors — those on the so-called *Web-safe* palette — are standard browser colors (meaning that the browser can display them properly). The Web-safe color palette is a set of 216 colors that are found within both the Mac and the PC system palettes. Because they are shared, you know that images that use these colors will look the same on both Macs and PCs without any dithering. For example, if you create an image that uses just four colors, you should use Web-safe colors. Otherwise, you run the risk of the browser using two similar colors side by side in a pointillism-like arrangement to visually approximate your custom color. Of course, this concern is quickly vaporizing because most monitors today are capable of displaying more than 256 colors at a time.

Reducing an image's bit depth

When you reduce an image's palette, you are reducing the bit depth of that image. A 24-bit image can use millions of colors if necessary, and an 8-bit image has only 256 colors to work with. What happens if you choose fewer than 256 colors? What's the bit depth then? Table 9-1 is a handy chart that equates bit depth to the number of colors in the palette. For you techies, it's a simple binary formula of 1 bit equaling two possibilities — black (off) or white (on).

Table 9-1	Color Bit Depth	
Bit Depth	*The Math*	*Number of Colors in Palette*
1 bit	2^1	2: black (off) or white (on)
2 bit	2^2	4
3 bit	2^3	8
4 bit	2^4	16
5 bit	2^5	32
6 bit	2^6	64
7 bit	2^7	128
8 bit	2^8	256

The numbers in Table 9-1 illustrate the *maximum* number of colors that each bit depth allows. When you reduce the number of colors in your image to keep the file size small, you can choose any number of colors you like. For example, you can choose a palette of 200 colors. The image is still an 8-bit, but because it uses only 200 colors, the file is smaller than if you used all 256 colors.

Incidentally, I show this chart going only to 8-bit color depth because GIF images must be 8 bits or less. The JPEG format, on the other hand, always compresses a 24-bit image.

If you are ever given the option to save your image as a 32-bit image, this means that you can save an *alpha channel* mask along with the 24-bit image. An alpha channel is a grayscale mask that controls an image's transparency. Where the mask is black, the image is completely masked and appears transparent. Where the mask is white, the image is completely opaque. Gray colors in the mask make the image appear at varying degrees of transparency.

The extra 8 bits for the alpha channel give you 256 levels of transparency control. So far, the only Web graphics format that supports alpha channel transparency is the PNG (Portable Network Graphics) file type, pronounced *ping*. The problem is that at the time of this writing, most Web browsers still don't support this level of PNG functionality with any reliability.

All the most popular Web graphics tools, such as Photoshop and Fireworks, allow you to reduce an image's palette from millions of colors to just a few colors. If you do this, however, you can't go back. After you remove colors from an image, the colors are gone forever — unless you use the Undo feature right away. You can't use Photoshop or Fireworks to convert the image back to 24-bit, and expect the millions of colors to magically reappear. The computer has to guess where to insert the lost colors, a process called *interpolation*.

To prevent the disaster of losing all these colors, always work in the highest bit depth (24-bit) when building Web graphics and then save your original source files separately from your finished graphics. (When you work in Photoshop and Fireworks, the default color mode is 24-bit, so you don't have to do anything special to ensure that you're working at the highest bit depth.) Reducing the palette down to 256 or fewer colors is the last thing you should do before exporting finished art for the Web.

Color palettes

After you have a good understanding of color bit depth and how it relates to the number of colors in the palette, your next step is to look into the kind of palettes you can choose from for your design. In addition to some pre-fab palettes, you can also create your own custom palette. However, on 8-bit monitors, only 216 colors display the same on all Web browsers without dithering — these are the Web-safe colors. Whatever palette you choose, if designing for older 8-bit monitor displays, it's a good idea to include as many of these 216 colors as possible.

The following sections detail some palette options.

System palettes

Macintosh and Windows each have their own pre-fab palette of 256 colors. The palettes are pretty generic. They include reds, oranges, blues — and everything in between to do a fair job of dithering a 24-bit image.

Take a look at the Macintosh's 256-color system palette in Figure 9-4. The palette of 256 colors actually contains all 216 Web-safe colors. The extra 40 colors (an assortment of blues, reds, greens, and grays) that appear dithered in the first two rows are not Web-safe.
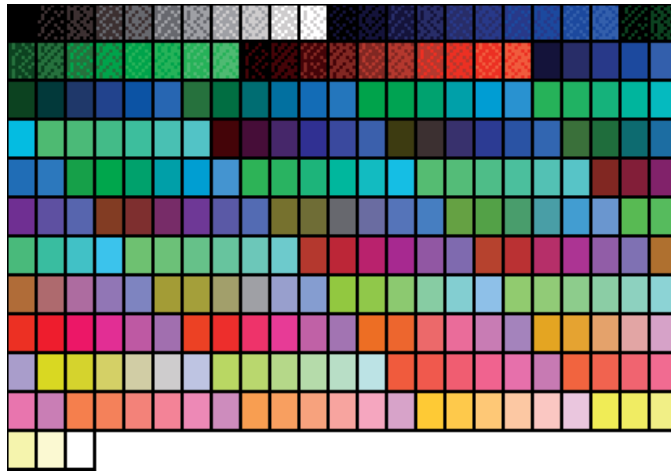
Figure 9-4: The Macintosh system palette of 256 colors.

The problem with forcing an image into one of these system palettes, however, is that the image looks highly dithered, as shown in Figure 9-5, because not all the colors in the palette are appropriate for the image. For example, all the pinks, reds, and oranges in the palette are not useful for dithering a seascape image.

Adaptive palettes

An *adaptive* palette is a custom selection of colors that are best suited, or adapted, for the image. For example, an adaptive palette for a seascape image uses a lot of blues and greens — not pinks and reds. An adaptive palette yields better results
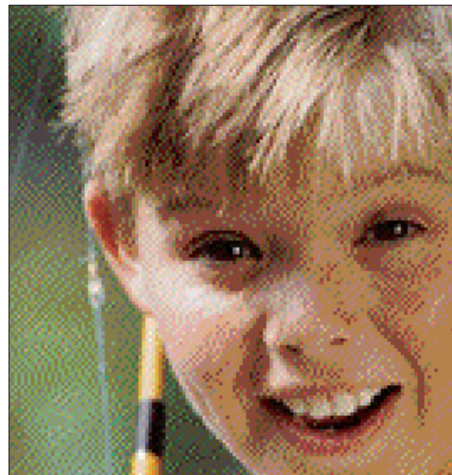
Figure 9-5: This image is pixilated because the 8-bit Macintosh system palette is being used.

because more of the 256 color slots go to colors that the image actually needs. This is my favorite type of palette because you can reduce the file size and still retain nearly the quality of a 24-bit image.

Compare the image qualities in Figure 9-6. The left image is the original 24-bit image. The right image uses an 8-bit adaptive palette but, in terms of quality, looks almost identical to the original. Keep in mind, however, that using an adaptive palette means that end users with 8-bit monitors see a dithered image because you are not using the Web-safe colors (see the next section). Most users nowadays, however, have 16- and 24-bit monitor displays, so adaptive palettes look great and are what I recommend you use for all GIF images.

Figure 9-6: Even though the left image is 24-bit and the right image is 8-bit, the quality of the two images is nearly the same.

Web-safe palette

The problem with using the pre-fab system palette or an adaptive palette is that you have no guarantee of what the image looks like on an 8-bit monitor. On an 8-bit monitor, browsers have their own color palette and force all images — 24 bit or otherwise — into this palette. Of the 256 colors in the browser palette, only 216 of them are the same on PC and Mac. So, if you're designing sites for people who might have older 8-bit monitors (government, education to name a few), you should use these 216 colors whenever possible. Fill your backgrounds with them, use them to color your text, and use them for any graphical elements.

Web-adaptive palette

Although an adaptive palette provides the best quality, you run the risk of the image looking bad on an 8-bit monitor. A middle ground is to use a Web-adaptive palette. Like an adaptive palette, this palette finds colors that are best suited for the image. The difference is that wherever possible, the colors snap to the nearest Web-safe equivalent. This kind of palette gives you the quality of an adaptive palette with more predictable display results on 8-bit monitors.

Give it a try

TIP

In a program like Fireworks or Photoshop, start with a 24-bit image and export it a few times, each time with a different aforementioned palette. Then set your monitor to 256 colors and open the GIF in a Web browser to see what it looks like.

Deciphering the hexadecimal color code

When you specify colors in HTML code, you don't use their RGB numbers. Instead, you use something called a *hexadecimal code*. For example, plain old white is 255,255,255 in RGB, but it is #FFFFFF in hexadecimal code. In hexadecimal code, the first two digits define the red color, the next two digits define the green, and the last two digits define the blue. Using the #FFFFFF example, the first two digits FF equal 255. So, this color is R=255, G=255, and B=255. All three colors at full power (255) make white.

Imagine that you are designing a Web site for a company that has a handful of company colors that they want you to use throughout the site. Instead of using a super-duper decoder ring to translate RGB values into hexadecimal code and vice versa, you can retrieve the code in both Photoshop and Fireworks. Otherwise, you really do need a special hexadecimal calculator (. . . and they do exist online!).

Here's how to use Fireworks to find the hexadecimal number of a sample color:

1. **Launch Fireworks.**

2. **Choose File➪Open from the menu bar.**

   Navigate to an image you want to sample color from and open it in Fireworks.

3. **Sample the color by clicking the Fill color swatch at the bottom of the Tools panel next to the fill bucket icon.**

   As shown in Figure 9-7, a pop-up palette appears assuming you want to pick a color (from the Web-safe colors), and your cursor turns into an eyedropper.

Hexadecimal code readout

Fill color swatch
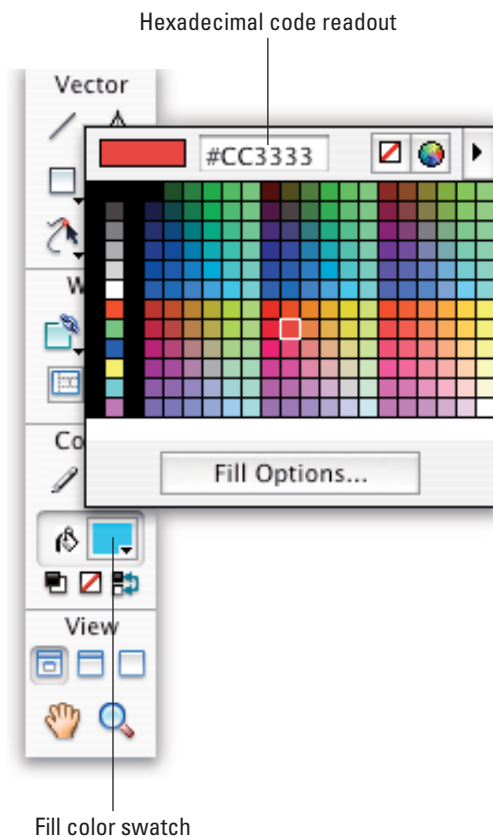
Figure 9-7: The Fireworks Fill and Stroke tools in the Tools panel allow you to sample colors in a document to find their hexadecimal values.

 4. **Move your cursor around in the palette and notice the hexadecimal readout updates as you roll over each color.**

 5. **Move your cursor out over your image and roll around and then look at each hexadecimal value in the readout.**

 The readout tells you the hexadecimal value of colors in your document — and even in the Fireworks interface components or on your desktop if you roll over them! Clicking samples the color and makes it your fill color.

Smart Web Color Usage

As you work in a graphics program to develop graphics, you can perform a number of actions to ensure that you get the best image quality at the lowest file sizes — more bang for the buck. Your color choices for text and graphical elements, the way you draw gradient blends, and the number of colors that you choose all affect the final outcome. The following sections present tips and techniques to help you create great-looking Web graphics (on all browsers and computers) while keeping the file sizes down for speedier delivery.

Use flat-colored graphics

Flat-colored graphics are ones that use a single solid color with no blends, gradations, or textures. For example, text, graphical shapes and illustrations, cartoon characters, and backgrounds that are filled with one solid color are flat-colored graphics. While you can incorporate textures and blends in part of the graphic, if most of it is a flat color, your compression is excellent, and yields a small file size. If 8-bit monitors are a concern, make sure you use a Web-safe color to fill all flat-colored areas. Figure 9-8 shows an example of a graphic that uses mostly text and flat-colored backgrounds. Though the image has some photography, it compresses to a tidy 15K.
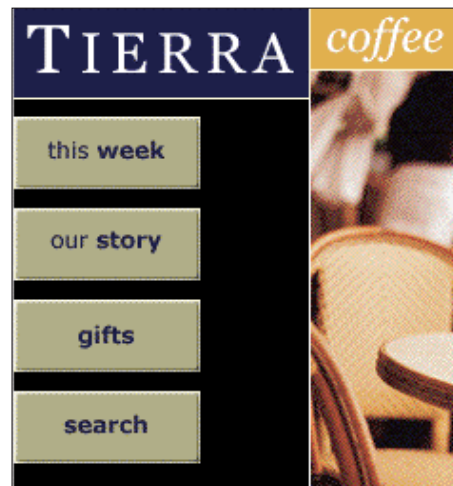
Figure 9-8: This graphic compresses well because most of it is filled with a single flat color.

Use as few colors as possible

Using fewer colors in a graphic creates a smaller file size with better quality. If, for example, you're saving the graphic as a GIF and therefore need to reduce its color palette to 256 or less colors, you can most likely get away with an adaptive palette of just 60 or so colors. If your design uses a ton of colors from all over the rainbow, you'll have a hard time defining a small palette — you simply need more colors to draw the design with any quality.

The design in Figure 9-9 uses a lot of different colors, not to mention the colored text and graphical areas. When I reduce the image to just four colors, making the file size just 10K, the whole design is ruined. To make this design work, I have to use at least 100 colors. This makes the file 40K. In this case, I'd be better off slicing the image into two pieces — a top text area and a bottom photo area. This way, I can save the top text area as a four color GIF and the bottom photo area as a JPEG. For more information on choosing the right file format for a particular image, see Chapter 12.

Image saved with
palette of 100 colors

Image saved with
palette of 4 colors

Figure 9-9: This design uses so many colors that it's impossible to reduce the palette to less than 100 colors and still look good. In this case, you are better off splitting it into two images, saving each one in a different file format.

Beware of the gradient blend

A *gradient* is a gradual transition from one color to another. For example, on one end, the image is red, and on the other end, it's blue. Gradients have the potential to really bloat the size of your file. If you're going to save the file as a GIF, make sure that the gradient blend goes from top to bottom rather than from left to right, as shown in Figure 9-10.

The type of gradient makes a difference because of the way the GIF format compresses an image. The GIF format reads each horizontal row of pixels in your image and records the color changes. In a top-to-bottom gradient, each horizontal row has the same color of pixels. In a left-to-right gradient, each pixel's color changes as you go across the horizontal row.
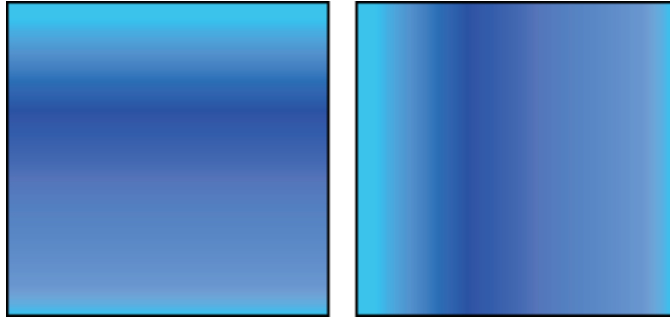
Figure 9-10: A top-to-bottom gradient blend compresses better than a left-to-right blend.

When an image has a mix of photos and flat colors, save it as a GIF

When a graphic has a mix of flat-colored art, text, and photographs, it's best to save the graphic as an adaptive palette GIF. JPEG often blurs your text and screws up your flat graphics by introducing *artifacts* — little granular intrusions. Both Photoshop and Fireworks allow you to preview your file format and color palette settings before you commit to one. This feature allows you to try a few different options because there are always exceptions to these guidelines. I like to try both JPEG 85% quality and Adaptive palette GIF 128 colors, adjust from there, and decide which route looks better and gives me better file size.

10

Building Web Graphics from the Ground Up

In This Chapter

▶ Exploring the difference between vector and bitmap graphics

▶ Understanding monitor and image resolutions for the Web

▶ Using tools like Fireworks and Photoshop to create Web graphics

▶ Integrating stock and digital photo images

▶ Scanning images for the Web

*J*ust when you thought you could get away with a completely bland, text-only Web site, reality and peer pressure set in. Today's Web sites are chock-full of eye-popping graphical coolness. To keep up with the Joneses and, as studies have shown, increase the usability and credibility of your site, your visual design must sparkle with the best of them.

After you come to terms with this basic fact of Web life, how do the artistically challenged proceed? A good place to start: Become proficient in the rules, tools, and techniques that surround the creation of Web graphics. Mastering the skills is half the battle. The other half is tapping your own creativity. Good design training doesn't hurt of course.

In this chapter, I discuss some basic issues surrounding Web graphics, I take a look at the leading software tools, and I walk you through techniques for churning out professional-quality graphics. Whether you're a seasoned designer from the print world or completely new to design, with a little practice you'll be cooking up Web graphics juicy — or cheesy — enough to eat.
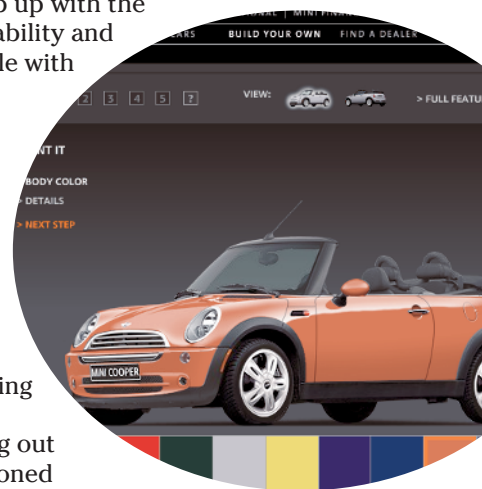
Bitmap versus Vector Graphics

All graphics that you encounter on the Web are either bitmap or vector graphics. The basic difference between the two is how they are drawn on your screen. This simple detail affects everything from image resolution to file size and format.

Bitmaps: A fabric of pixels

When explaining bitmap images, I can't help but think of my old childhood toy, the Lite Brite. Those of you who were pampered enough as children to play with this toy remember that you plugged little colored pegs into a grid and then flipped the switch to light them up.

Bitmaps work pretty much the same way. In simplistic terms, a bitmap graphic is a grid filled with tiny colored pixels, as shown in Figure 10-1. To draw a bitmap graphic on-screen, the computer lays out a grid, say 100 by 100 pixels, and then *maps* a color to each individual pixel. That's 10,000 pixels to draw!
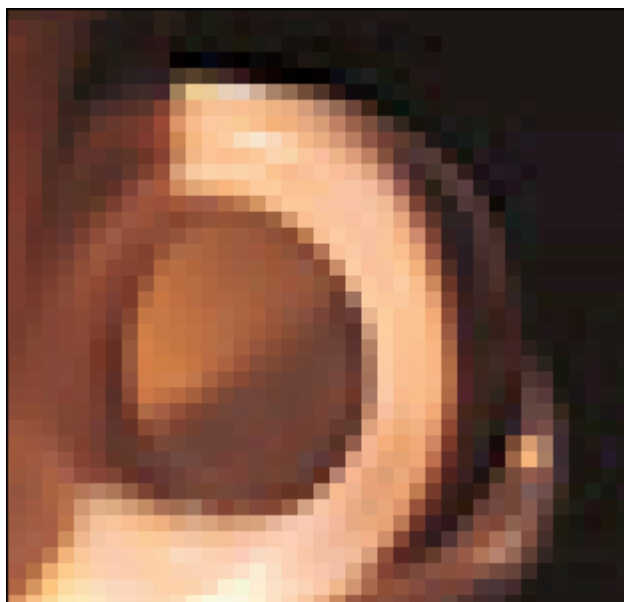
Figure 10-1: Zooming in on a bitmap reveals hundreds of colored pixels all working together to create an image.

Vectors: For the mathematically inclined

If you don't remember the Lite Brite, then surely you remember "Connect the Dots." Vector graphics employ a similar strategy. A mathematical formula places *points* on the screen and then connects them with *paths*. For example, to draw a triangle-shaped vector graphic on-screen, the computer simply lays down three points, connects them, and then fills them with a color. In Figure 10-2, you can see a handful of points — some with handles coming off of them. These handles control the curve of the path in between two points.
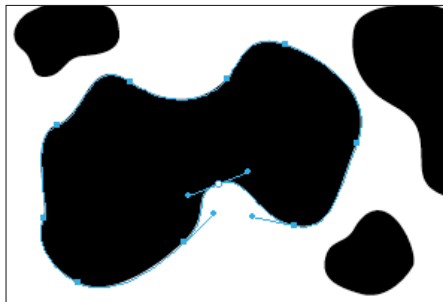
Figure 10-2: Vector images are a leaner brand of graphics defined by a series of points connected by curves.

If you're thinking that vectors can draw graphics on-screen far more efficiently than bitmaps, you're right. Vector graphics have extremely small file sizes, making them ideal for online delivery. Both vector and bitmap graphics, however, have their pros and cons.

The vector-bitmap showdown

As a rising Web graphics star, you should know a few things about the ups and downs of vectors and bitmaps before you push your first pixel:

✔ **Bitmaps are highly prevalent on the Web.** All JPEG, GIF, and PNG graphics that you encounter on the Web are bitmaps. Ironically, many of these graphics began life as vectors before they were converted into bitmaps, as shown in Figure 10-3. Why? Because drawing graphics and interesting effects with vectors is often faster and easier than creating the same effect with bitmap graphics. It's much easier to make changes to vectors because you just move a point and the line follows. To make changes to bitmaps, you often need to redraw them. To become GIFs and JPEGs, vector graphics must be converted to bitmaps.

GIF (depending on whom you talk to) is pronounced either as it sounds — giff (my personal favorite), or *jiff* (as in Jiffy Pop popcorn), and stands for *Graphics Interchange Format*. JPEG, pronounced *jay-peg*, stands for *Joint Photographers Experts Group*. PNG, pronounced *ping*, stands for *Portable Network Graphics*.

✔ **Bitmaps are supported by more formats.** To date, the only standard Web format that supports vector graphics is the Flash SWF format (pronounced *swif*). Generally, the SWF format is used for making interactive

animated movies that you can plug into your Web page. The format isn't used for saving static graphics (although you certainly can if you want). To view this format, however, your browser must have the Flash plug-in installed (which is a free download).
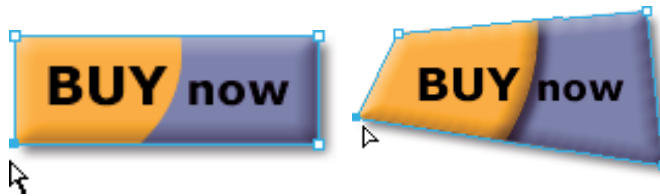
Figure 10-3: It's easy to change the shape of this vector-drawn button by moving its points.

✔ **Bitmaps can't maintain quality if they are resized.** Bitmap graphics have fixed resolutions (see the section "Image resolution" later in this chapter for more information on resolutions). As I state earlier in this chapter, bitmap graphics are laid out in a grid of pixels. Like ½-inch grade chicken wire, this grid is a fixed size. And like chicken wire, if you stretch the image, you end up with a contorted mess. So, avoid using the HTML height and width tags to squash and stretch bitmap images to fit your Web page. Make your images the correct size in the first place.

✔ **Vector graphics can easily be resized.** Vector graphics, on the other hand, are resolution-independent. Because they're drawn according to a mathematical formula of placing points on the screen, the *grid size* (resolution) doesn't matter. This means that you can infinitely squish and stretch vector graphics up, down, and side to side without losing quality: They are the Play-Doh of the Web. Take a look at the contorted, albeit high-quality, icon that I made by stretching a vector illustration in Figure 10-4.

Figure 10-4: You can enlarge and stretch vector graphics without losing image quality.

For a quick reference chart of all the bitmap and vector Web graphic formats that you'll deal with, take a look at Table 10-1.

Table 10-1		The Scoop on Graphic Formats	
Format	*Full Name*	*Bitmap or Vector*	*Best Uses*
GIF	Graphics Interchange Format	Bitmap	Graphics like cartoons with a lot of solid-colored areas, or those with a lot of text elements.
JPEG	Joint Photographers Expert Group	Bitmap	Photographic images or images with a lot of blending colors (a rainbow, for example).
PNG	Portable Network Graphic	Bitmap	Images that have both photographic-like areas and solid-colored areas.
SWF	Shockwave Flash	Vector	Illustrated images (not photographic). Smooth motion animation.

Monitor resolution

If you've ever used a computer, you probably realize how clear the monitor is compared to your TV. Really, who can read all the legalese fine print that scrolls by in that "0% down payment" car commercial? That same fine print is a cinch to read on your computer screen — and not just because you're sitting so darn close to the screen. You can see clearly because your computer monitor has a finer resolution than your household tube.

Resolution refers to the number of pixels squeezed into a linear inch. Standard computer screen resolutions vary from 72 to 96 ppi (pixels per inch) — that's a lot of detail. Your TV, on the other hand, is nearly half of that, and because most broadcast TV is analog and not digital like your computer, the image is all blurred together.

Image resolution

Why should you give a hoot about monitor resolution? Because the monitor is your Web design canvas. Although the monitor is a fixed resolution (after all, it is a piece of hardware), graphics come in varying resolutions. In Web design, your graphics must *match* the screen's resolution.

Although monitors vary from 72 to 96 ppi, the Web graphics standard is 72 ppi. The professional lingo is *72 dpi*, which stands for *dots per inch,* a carry-over from the print production days.

Here's a frequently asked question: Won't a higher resolution image look better on my Web page? The answer is no, because even though a 300 dpi image looks great in a printed piece, the fixed 72 to 96 dpi monitor isn't capable of showing all this detail. Regardless of the image's resolution, the browser simply shows it by its physical pixel dimension, say 200 pixels wide by 200 pixels high.

Both images in Figure 10-5 are 247 x 167 pixels. Because the left image is 150 dpi, each dot is smaller, so more dots can be squeezed into an inch. That's why the whole image shrinks when you print it. The computer monitor, however, isn't capable of displaying such tiny dots, so it blows each dot up to 72 dpi size, as the example on the right shows. So, you're back where you started — higher resolution images don't improve the quality. Ergo, it's best to just stick with the default 72 dpi resolution in the first place.
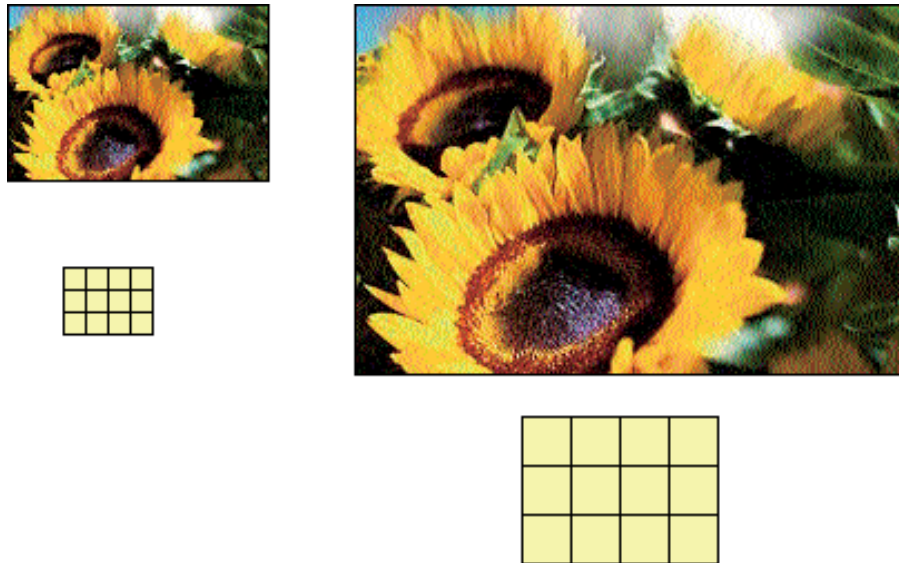
Figure 10-5: High-resolution images have tiny dot sizes — much smaller than a computer pixel.

The Usual Software Suspects

Before you start making Web graphics, you should become familiar with the most-used Web graphic software tools. Whether you're the weekend warrior type or the sun-tanned professional, someone has the Web design tool for you. Keep in mind that several tools are available on the market for pushing pixels around, so I just list the usual suspects.

Adobe Photoshop

www.adobe.com/products/photoshop

For creating, editing, and manipulating bitmaps, no other tool on the market holds a candle to Photoshop. The problem with Photoshop, however, is that learning to use it to its fullest is akin to learning to fly a jumbo jet. In addition to the years it may take to fully master Photoshop's power, it also comes with a steep price tag. A less expensive alternative to Photoshop is Adobe Photoshop Elements. Elements has the most commonly needed and used features of Photoshop and the same user interface. It's an excellent program to get started with.

Macromedia Fireworks

www.macromedia.com/software/fireworks

Not only is Fireworks a fantastic Web graphic creation tool, but it also empowers us non-techies to add interactivity to our graphics — for example, rollover buttons and drop-down menus. Fireworks is also much less expensive than Photoshop. The one downside to Fireworks is that its bitmap creation and manipulation abilities are not as robust as Photoshop's. You can, however, import layered Photoshop files, fold them into your Web page layout, and export Web-enabled graphics and code.

Fireworks is actually my tool of choice for building Web page designs, using Photoshop only to build complex graphical components that I then incorporate into my Fireworks layout. Fireworks has the added flexibility of being able to export Photoshop layered files. While you can go back and forth between the two programs, however, be aware that you'll need to do some clean up because not all filters, effects, and text treatments are preserved.

Paint Shop Pro

www.corel.com

Paint Shop Pro is a cheaper alternative to Fireworks for creating bitmap and vector graphics and then optimizing them for Web delivery. Paint Shop Pro also enables you to add links and rollover buttons to your designs. The downside is that it works only with Windows. Because many professional Web graphic production teams use a mix of Mac and Windows platforms, Paint Shop Pro files are not easily passed from one team member to the next.

Adobe Illustrator and Macromedia Freehand

```
www.adobe.com/products/illustrator
www.macromedia.com/software/freehand
```

These two programs create vector graphics that you can import into other programs like Photoshop, Fireworks, and Flash to prepare them for the Web (export them as GIF and JPEG files, and then add interactivity like links and rollover buttons). In other words, although these programs are great for building complex vector graphic components, they are not the tools used to design the full Web page layout.

Macromedia Flash

```
www.macromedia.com/software/flash
```

Flash allows you to create interactive Web applications with its robust scripting language ActionScript, and design animations that leverage the tight file sizes and scalability of vector graphics. Flash files are output as SWF files such as the one shown in Figure 10-6, an animated Flash Christmas card from JUXT Interactive (`www.juxtinteractive.com`). The slight downside to Flash is that it requires end users to have the proper Flash plug-in and version installed on their browser to view (although something like 98 percent of people have some version installed).

www.juxtinteractive.com

Figure 10-6: Flash uses vector graphics to produce fluid animations that download quickly.

Pixel-Pushing 101: Creating a Banner

After you know just enough to be dangerous, it's time to roll up your sleeves and design a common Web page element — a banner for the top of the page. Not only does this exercise allow you to get some practice with creating actual Web graphics, but you also get to put on your visual and user interface design thinking cap.

A common graphical element found on the Web is a banner ad. Ads come in a variety of sizes, but what I see a lot of these days are ads that go on the side.

In this section, you create a simple ad banner. Don't expect a design master-piece. Just think of this as a good way to get your feet wet with Web design software. In the following steps, I walk you through the process of creating a Web banner using Fireworks. Photoshop offers similar capabilities, but the process is pretty simple in Fireworks.

Follow these steps to create a banner in Fireworks:

1. **Launch Fireworks and choose File⇨New from the menu bar.**

   The New Document dialog box appears.

2. **Set the dimensions for the banner.**

   Set the file's dimensions to 185 pixels wide x 130 pixels high. Leave the screen resolution at 72 dpi, the standard Web image resolution.

3. **To set the Canvas color, click the Custom option. Then click the color swatch to the right to pick a new color, as shown in Figure 10-7.**
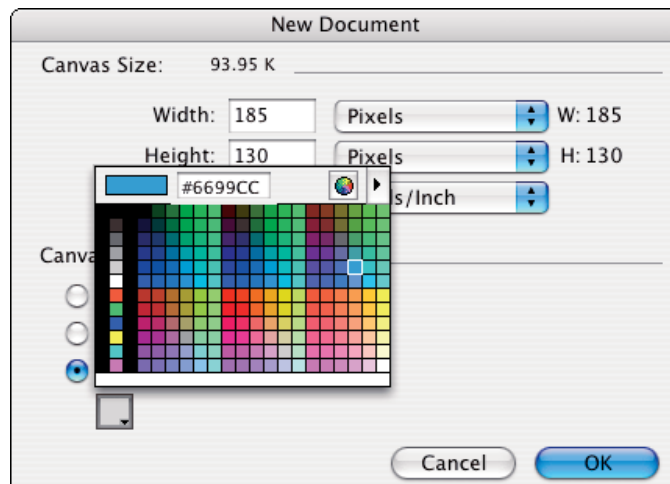
Figure 10-7: Choose a custom background color for your design.

4. Click OK to close the dialog box.

5. Select the Text tool (as shown in Figure 10-8), click once in the upper-left corner of the document, and begin typing your message.

6. When you finish typing, switch to the Pointer tool in the upper left of the Tools panel.

The text object is selected in the document.

7. While the text object is selected, make the following changes as desired:

- Use the Property Inspector (shown in Figure 10-9) to apply font, font size, and color settings to your message.

- Use the Pointer tool to move it around.

I embellished the text by adding a few rectangles filled with gradients, some white thin lines, among other details. The whole design is built with vector graphics so that I have utmost flexibility.

If you want to edit your text, double-click it to use the Text tool. To exit editing mode, select the Pointer tool again.

8. If you're feeling really adventurous, you can import your company logo or product image. To do so, choose File⇨ Import. Then in the Import window, locate your logo and click OK.

Your cursor now looks like a corner icon.

9. To place your logo or image, click once on the document.

When you place your image, you can click and *drag*. That way you dynamically resize the image as you import it.

10. To resize your logo to fit, select it with the Pointer tool. Then choose Modify⇨Transform⇨Scale. Grab one of the corner handles as shown in Figure 10-10, press and hold Shift (to constrain proportions), and drag to scale up or down. When you like the new size, press the Return or Enter key.

Remember, as discussed in Chapter 9, scaling a graphic up results in a blurred image.

After you create the text object(s) and import an image, Fireworks treats them all as separate objects. To move them around on the page, choose the Pointer tool, and click the object to move it.

Pointer tool

Tools panel

Text tool
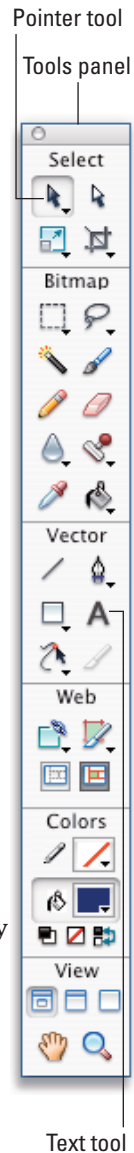
Figure 10-8: The Tools panel.

Choose font size

Property Inspector　　　　　Select font　　　Pick a text color
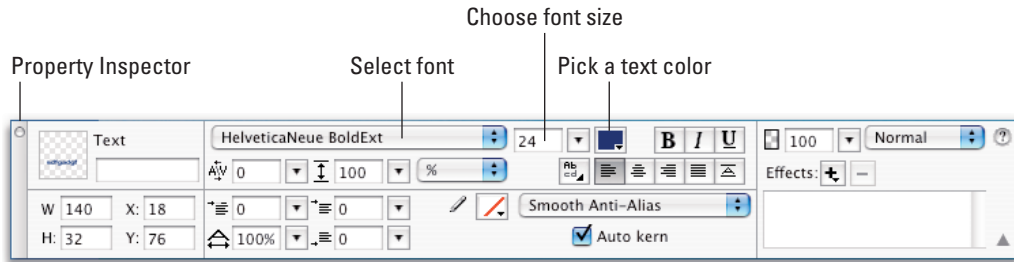
Figure 10-9: Apply font, size, and color settings in the Property Inspector.

Figure 10-10: To scale an image, click and drag one of the square corner handles.

11. When you're satisfied with your new banner, choose File⇨Save.

TIP

Just like in Illustrator or Freehand, you can change the *stacking* order of your objects. For example, if one object is behind another and you want it to be on top, select it with the Pointer tool and choose Modify⇨Arrange⇨Bring to Front.

Congratulations! You've just created your first official *programmer art* — the technical term for cheesy Web graphics. As you can see, you've only brushed the tip of the iceberg of what programs like Fireworks can do for you. Just imagine the level of cheese that you can achieve if you really put to your mind to it!

Why Fireworks?

Fireworks is an ideal choice for creating Web graphics because after you finish building your graphics, you can easily add links, rollover buttons, and even drop-down menus without needing to write any code. When you export your Fireworks file, you get not only the graphics, but also the HTML page that makes it all work. Photoshop does have the ability to slice and optimize Web graphics, but it cannot add interactivity like this to the exported HTML page.

Image Manipulation

Of course, not all Web graphics consist entirely of buttons, illustrations, and text. Invariably, your Web page will have a few photos sprinkled throughout the layout. So how do you get a photo from a camera into your computer? After you do that, how do you fix it up and resize it to fit your layout?

The first issue has been somewhat resolved by the recent proliferation of digital cameras. Now, even amateurs can take high-quality images and get the images into the computer. With regular cameras, you must scan the images. Another route is to use *stock* photography and illustration: These are professional photos, already digitized, that you can purchase online with your credit card. Although nothing beats the convenience of stock images, you can't always get the image you want at the price you want to pay. Royalty rates and license agreements vary greatly between stock image companies. Even prices can vary from image to image within the same company, and you need to read the agreement carefully to make sure you have rights to use the image in various countries. The preeminent stock media company is Getty Images (`www.gettyone.com`). It offers both royalty-free and licensed images:

- ✔ *Royalty-free* means you pay one flat fee for a particular size of image and can use it for whatever purpose
- ✔ *Licensed* means you pay according to the type of use and length of use

After you have an image in the computer, the next step is massaging it to suit your layout's needs. This entails everything from simple color adjustments and resizing to custom collage work when you need to combine imagery together. Bar none, the best software application for this type of image manipulation is Photoshop.

Direct from digital cameras

So many digital cameras are on the market that I don't even want to begin naming brands and models; that is a book in and of itself. For more information, you can check out Julie Adair King's *Digital Photography For Dummies* or David Busch's *Digital SLR Cameras & Photography For Dummies,* both published by Wiley. When shopping for a digital camera, however, here are a few things to keep in mind:

- ✔ **Make sure it can take large photos.** Most cameras offer a few different image size settings because the bigger the image size, the fewer photos you can fit into the camera's memory. Also, most cameras capture images at 72 dpi. That's okay if the camera is capturing a large image — say 3,000 x 3,000 pixels or more — because you are capturing a lot of detail in that many pixels. This resolution is not okay if the image is small, say 500 x 500 pixels, because you won't get enough detail to work with.

REMEMBER

By capturing a larger image, you have more flexibility when you need to edit. It's always better to shrink bitmap images rather than blow them up. Because it's a bitmap, when you enlarge them, you distort and blur their quality.

✔ **Make sure it comes with a flash.** Or it should at least offer some other means to control lighting for indoor and outdoor situations. Some of the less expensive digital cameras give you no control over lighting, whereas the more expensive cameras allow you to take pictures at night.

Connecting digital cameras to your computer is becoming less of an issue. You can plug most modern cameras directly into newer computers using a Universal Serial Bus (USB) connection. An icon representing your camera shows up on your desktop just as with a CD or DVD. You can double-click the camera icon to open a folder of images and copy them right to your computer. Figure 10-11 is an example of an image I took with my Sony digital camera, plugged into my computer, and opened in Photoshop ready for editing.

Figure 10-11: Images right out of a digital camera are excellent quality with enough resolution for Web usage.

Scanning images

If you're using a regular camera with good old-fashioned film, you must scan the prints, transparencies, or slides to get them into the computer. Because the Web is a relatively low-resolution environment (compared to the print world), you don't need to work with super high-resolution images. Remember that

Web graphics should always be 72 dpi. The two primary options you have for scanning images for the Web are a *flatbed* scanner and a *drum* scanner:

- ✔ **Flatbed scanners** — sometimes called *desktop* scanners — are affordable enough to include in your home workstation. Flatbed scanners get their name from their big flat surfaces, which are sort of like a copier machine, where you place your photos to scan.

- ✔ **Drum scanners** are big fancy machines that only professional service places own. Drum scanners yield extremely high quality that is ideal for print world, but overkill for scanning Web graphics. For Web graphics, all you need is an affordable desktop flatbed scanner.

For desktop scanners, always scan at roughly twice the resolution you need. For Web graphics that need to be 72 dpi, for example, scan the images at 150 dpi. This way you capture enough detail to make editing easier. You can zoom into detail you need and crop away the rest, and you can always scale down, which often eliminates the graininess of scanned images.

Using stock photography and illustration

You can find virtually any kind of photo or illustration that you need for a project, and even video and audio clips, at one of the many online stock photo companies. For instance, you can go to my favorite, www.gettyone.com, and search through thousands of images. If you are a registered user (it's free to register), you can download *comps* that you can incorporate into your Web design layouts and show clients. Remember to keep the given names of the images you download. If the client likes the image, you can purchase it online by searching on its filename. Another online stock photography option is Corbis (www.corbis.com).

Each stock image usually comes in a few different sizes and resolutions. The smaller the image, the less it costs. Fortunately in Web design, you only need the smallest image (and by *small,* I mean 700 x 1000 pixels, which isn't that small). The price ranges according to the image. Royalty-free images cost about $25.00 to $50.00 for a 72 dpi, 800 x 800 pixel image. Royalty-free images are cheaper than licensed images, but they're often not as nice as the more expensive licensed images.

The special sauce: Digital editing

More often than not, an image fresh out of the camera or off the Internet needs a little editing before it's ready for your Web page layout. For one reason, it's probably not the right size, and second, it may contain some imperfections, need to be cropped, combined with another image or graphical element, or be in need of some embellishments.

The best tool on the market for editing bitmap images is Adobe Photoshop. Photoshop offers extensive control over every aspect of your image. The

problem, however, is that Photoshop is a very deep program. You may need a long time to figure out all of its power. For more information on Photoshop, take a look at *Photoshop CS2 For Dummies* by Peter Bauer or *Photoshop CS2 All-in-One Desk Reference For Dummies* by Barbara Obermeier.

In these next few steps, you scratch the surface of Photoshop's capabilities and discover some basic editing techniques, such as cropping, adjusting the exposure, resizing, and adding a soft, feathered edge:

1. **Launch Photoshop.**

2. **Choose File⇨Open.**

   Locate a photograph to work with. Choose a photo that needs cropping and a little work on the exposure.

3. **Crop the image.**

   Choose the Marquee tool in the toolbox and draw a box around the portion of the image that you want to keep, as shown in Figure 10-12. If you mess up, you can click once outside the selected area and start again. When you're satisfied with the selection, choose Image⇨Crop.
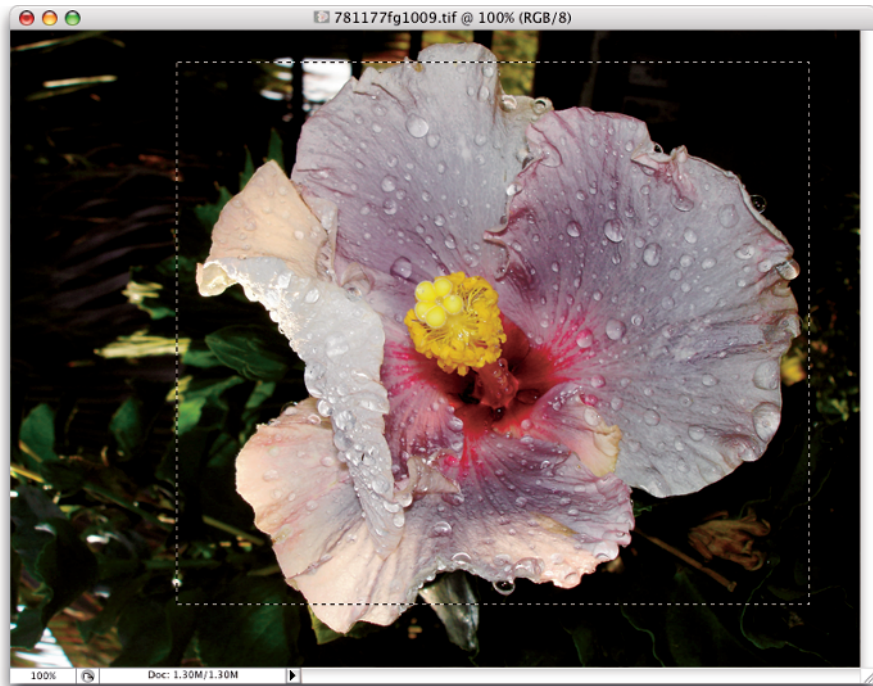
Figure 10-12: Make a selection around the area you want to keep.

4. Adjust the contrast.

To enhance the contrast of this photo, choose Image⇨Adjustments⇨ Levels. The Levels interface in Figure 10-13 shows the balance of lights and darks in your image. To increase the dark areas, drag the left triangle toward the center. To increase the light areas, drag the right triangle toward the center. Your image updates as you drag the sliders, so you can see the effects of your actions. Click OK.

Drag this triangle inward to increase dark areas.

Drag this triangle inward to increase light areas.
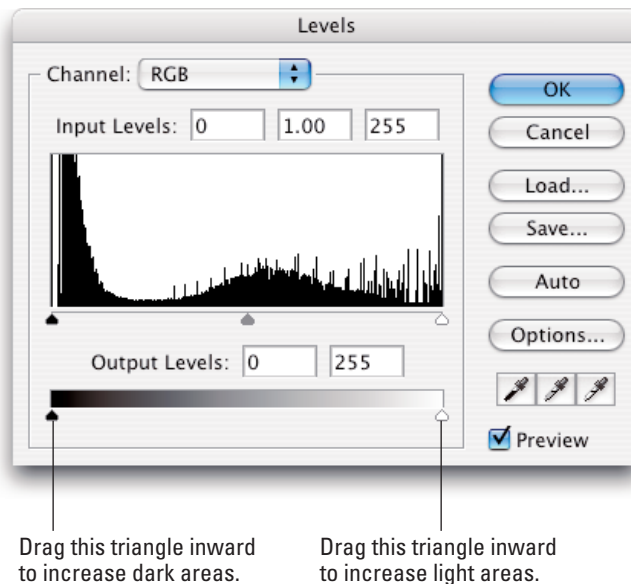
Figure 10-13: The Levels interface shows a visual diagram of your image's lights and darks.

5. Adjust the color.

You can use one of several tools in Photoshop to adjust the color balance of an image. For this example, I use the Hue/Saturation tool shown in Figure 10-14 to make the image monotone. Choose Image⇨Adjustments⇨ Hue/Saturation. To make an image monotone, check the Colorize option in the lower-right corner. The image becomes several shades of one color, such as purple, but you can change the color scheme by moving the Hue slider.

6. Resize the image.

To resize the image, first choose Select⇨All and then choose Edit⇨Free Transform. Handles appear at the corners of the image. Press and hold Shift (to retain proportions) and drag one of the corner handles inward

to shrink the image. When you reach the size you want, press Enter to shrink the image. Note that this technique resizes just the image, not the document. To resize the entire document, you would choose Image⇨ Image Size.
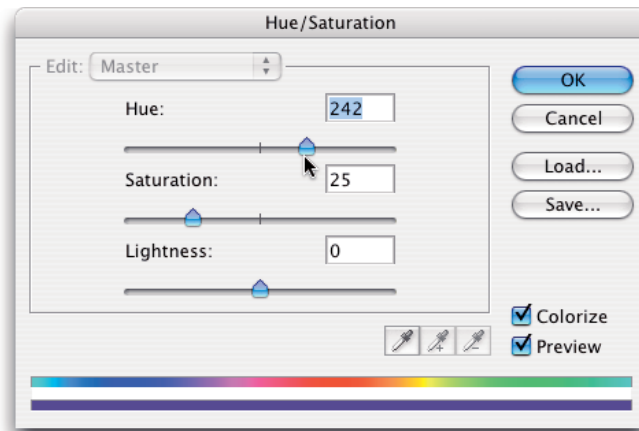
Figure 10-14: Use the Hue/Saturation tool to make an image monotone and shift the color scheme of an image.

TIP

You can also use the Free Transform tool to rotate your image. If you place your cursor just outside the corner handle, you see a curved two-way arrow. Click and drag to rotate.

7. Add a feathered edge.

To turn this image into an oval shape with a soft feathered edge, use the Elliptical Marquee tool. The Elliptical Marquee tool is hidden behind the square Marquee tool on the toolbar as shown in Figure 10-15. To access it, click and hold the square Marquee tool, and then select the Elliptical Marquee tool from the pop-up menu that appears.

Start drawing an oval-shaped selection in the middle of the image. Press and hold Alt or Shift to make a perfect circle. After you draw the selection, you can reposition it to make sure that it encompasses the correct part of your image. Simply click and drag anywhere inside the selection to move it.
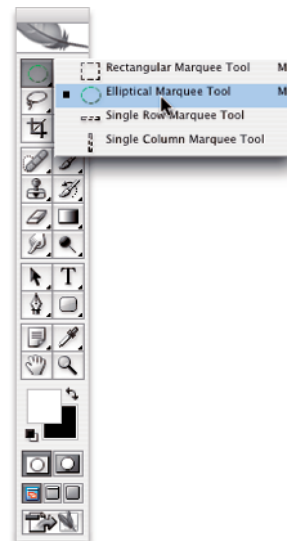
Figure 10-15: The Ellipse selection tool.

To soften the edges, choose Select➪Feather. In the Feather Selection dialog box, enter a Feather radius of 10 pixels and click OK. When you click OK, you won't notice any change in your selection, but believe me, it's feathered.

8. **Lift the image onto its own layer.**

To see the effects of your feathered edge, lift the image onto its own transparent layer. While the feathered selection is still active, press ⌘+J or Ctrl+J, or choose Layer➪ New➪Layer via Copy from the menu. In the Layers palette, you now have two layers (Layer 1 and Background), as shown in Figure 10-16. (If you don't see the Layers palette open, choose Window➪Layers.)

Hide your original Background layer by clicking once on the eye icon in the Layers palette. Now you can see the effects of your feathered edge as in Figure 10-17. The checkerboard background is an indication of transparency.

9. **Add the finishing touches.**

To polish up this image, replace the original image in the Background layer with a solid color. In the Layers palette, click the Background layer to make it active, and make sure the eye icon appears.

Click eye icon to turn layer visibility on and off
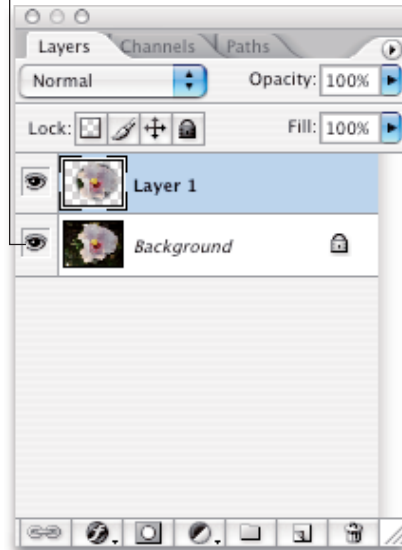
Figure 10-16: The Layers palette shows a copy with a soft feathered edge in the top layer.

Choose a new foreground color by clicking once on the top-most color swatch in the toolbar (you see two swatches stacked on top of each other — the top one is for selecting the foreground color, and the bottom one is for selecting the background color). When the Color Picker window opens, choose a new color and click OK. (Note that the Color Picker has a check box option to view only Web-safe colors.) To fill the Background layer with your new color, choose Edit➪Fill. Your background now fills with the new color.

Voilà! You now have an image like Figure 10-18 ready to save for your Web site.

Figure 10-17: In your document, you see a copy of your image with a soft edge.

Figure 10-18: The final result should look something like this.

10. **Export a Web-ready JPEG.**

    To save this image, choose File⇨Save for Web. On the right side of the window that opens, choose JPEG from the first menu. Next to Quality, click the default 60 number and adjust the percentage upwards and downwards to see which works best. Notice in the lower-left corner, you can see what the file size of the image will be and how long it will take to download. When finished, click the Save button. Name and save your new image.

11

Presenting Your Design Masterpiece

*A*fter a lot of planning, assembling a site map, and brushing up on a host of Web graphic design issues (everything from designing above the fold line to using type and color effectively), the next step is to put everything you know to good use. Before you can begin any graphic or HTML production on a Web site, you must first prepare a variety of design options and present them to the client for approval.

The designs should be mock-ups of a few finished Web pages so a client can get an idea of what the final site may look like. From these mock-ups, a client can choose the final direction of the site.

In this chapter, I show you how to prepare a presentation so clients can see how your designs look in a Web browser. I also show you how to print your designs and mount them on boards. Together, an online and a printed presentation helps clients better understand your design ideas, which enables them to choose a direction for their site. I also give you tips for pitching your work to clients and guiding them toward the design solution that you think is best.

[return to index]

Adomo Home Page

ROUND-1-01

p-1-1-01

1-1-02

adomo

Smooth

events

Developing Design Directions

To help clients visualize the final look and feel of the site, you should assemble a set of three to five different graphic design directions for them to choose from. A *design direction* is a complete graphical mock-up of a Web page — including all the HTML components, such as forms, text, and buttons. The idea is to visually show what the final page will look like without building the actual page in HTML. You should even go so far as to save the mock-up as one giant JPEG file, with 100% quality (no compression), and show it in the browser window. As the mock-up for `www.lopuck.com` in Figure 11-1 shows, this technique makes the page look even more real.

www.lopuck.com

Figure 11-1: If you make your design directions look real, the client can better choose a design.

Each design direction that you present should include a set of two pages — the home page and one sublevel page. This way, the client can see how the proposed graphic treatment works throughout the site, and it forces you, the designer, to ensure your design can work for the entire site. The designs should also show how the navigation works in the site. After you round up a few different designs, you can organize them into a polished presentation.

Getting design ideas

The big question: Where do you start? Staring at a blank white screen can be quite intimidating. The knowledge that you have to produce not just one but as many as three or more different designs is sure to conjure up the evil creative block that so often plagues writers. To ward off the block and start the creative juices flowing, I often look outward, not inward, for ideas. Here are three places to look:

- ✔ **Ask the client.** One of the simplest ways to find inspiration is to ask the client for ideas. To better understand the desires and expectations of your clients, ask them to provide you with a list of Web sites that they like (and sites that they don't like) and why. This helps shape your creative thinking toward something that the client ultimately appreciates.

- ✔ **Look online.** Another source of inspiration is to look online at various award-winning Web sites. In fact, always be on the lookout for interesting Web sites and bookmark them for future reference.

  A few sites hand out awards for various kinds of Web sites. These sites are one-stop idea shops that showcase different kinds of sites on a continual basis. As shown in Figure 11-2, one of my favorites is the Communication Art's Web site at www.designinteract.com. This site features a Site of the Week archive, and hosts an annual interactive design competition covering a number of categories from e-commerce to self-promotion. Plus, they maintain an archive of past year's winners for a truly expansive look at a variety of excellent Web site examples.

- ✔ **Look at the client's existing materials.** An obvious and crucial place to look for inspiration is the client's current set of creative material. Most companies probably have a number of things from which you can base your designs. Design elements from the company's products and packaging design, marketing brochures, office interiors, and logo are great themes to incorporate into your design directions.

Ideally, you shouldn't be the only one developing all three or more of the design directions for a client. After you do the first design, regrouping and coming up with two more unique designs is very difficult. The best scenario is to delegate one set of design directions to a different designer. This way, you get a wide variety of solutions that don't just look like variations of the same design.

Figure 11-2: Online Site of the Week showcases are a great place to find a lot of ideas in one convenient location.

If you are a one-man show and can't afford to have multiple designers each working on a different design, you have to flex some creative muscle to come up with all the designs yourself, but you can do it. One idea is to think of different themes to explore. For instance, you can explore a colorful geometric theme in one design and a modern, monotone photographic theme in another. By planning a few different themes, you can stay focused and find unique solutions for each design.

Integrating the venerable brand

When designing your mock-ups, you must pay careful attention to how you integrate the client's branding. For those of you new to design, the *brand* is not just a company's logo and colors. It's also a company's image — the way it wants a customer to perceive the company. For instance, Starbucks is not

just a place to buy coffee — it's an experience. It's a place for yuppity types to commingle and enjoy one another's company while sipping the world's best — or supposed best — coffee.

A lot of established companies have brand guidelines that you should ask for. These guidelines contain often stringent rules for using the company's logo, colors, photography, and typography in all manner of creative applications. You need to read through these carefully.

TIP

The designs you produce must not only follow the brand guidelines, they must also exude the company's image and attitude through the way you decide to apply the company's colors, fonts, and imagery. The easiest way to guide your design choices while you work is to always keep the company's target customer in mind. For example, if you were to design a site for Starbucks, think about what kind of design would appeal to the kind of folks who call coffee "lattes" and "frappaccinos."

Designing treatments for the home and subpage

As I state earlier in this section, each design direction should contain two pages: the home page and one subpage. The home page is important to show because it is the first page visitors see. The subpage is important to include because it shows how the design themes on the home page carry through to the lower level pages. The subpage also shows how users navigate through-out the site — an important detail for enabling the client to make an informed decision.

To build each mock-up, I like to start in a Web graphics program, such as Fireworks or Photoshop, and create a new file that is the size of the final Web page. A good dimension to choose is 800 x 600 pixels — the viewable area of most browsers. Also, set the background color to the color that you expect to use in the final Web page. Keep in mind that each design direction may have a different background color. This simple change can make a big difference in the overall feel of the direction.

REMEMBER

Your mock-up should look as real as possible — even showing graphical renditions of the HTML components. You're better off mocking up the design in a graphics program rather than building it as a true HTML page because you can lay out the design faster graphically. At this point, your focus is on generating as many design ideas as possible — not spending time implementing a Web page that the client may not choose.

To graphically create HTML parts that look real, I like to use Web authoring software, such as Dreamweaver, to quickly make buttons and fields with the right labels and sizes. Because I'm just making button elements and not laying out an entire page, the process is fast. After I build the HTML parts I need, I take a screen capture of them. Then, it's a simple matter of pasting the screenshots into my mock-up. Here's how you do it:

1. **Launch Dreamweaver.**

2. **Choose File⇨New to start a new Web page document. Choose Basic Page from the Category field and HTML from the Basic Page field. Click OK.**

3. **Set the background color.**

   If your Web page mock-up uses a background color other than white, you should set Dreamweaver's background to match. To change the background color, choose Modify⇨Page Properties. In the dialog box that appears, click the color swatch next to Background and choose a new color from the pop-up palette, as shown in Figure 11-3. For this example, I keep it simple and use the default white background color.

Click this swatch to set a background color.
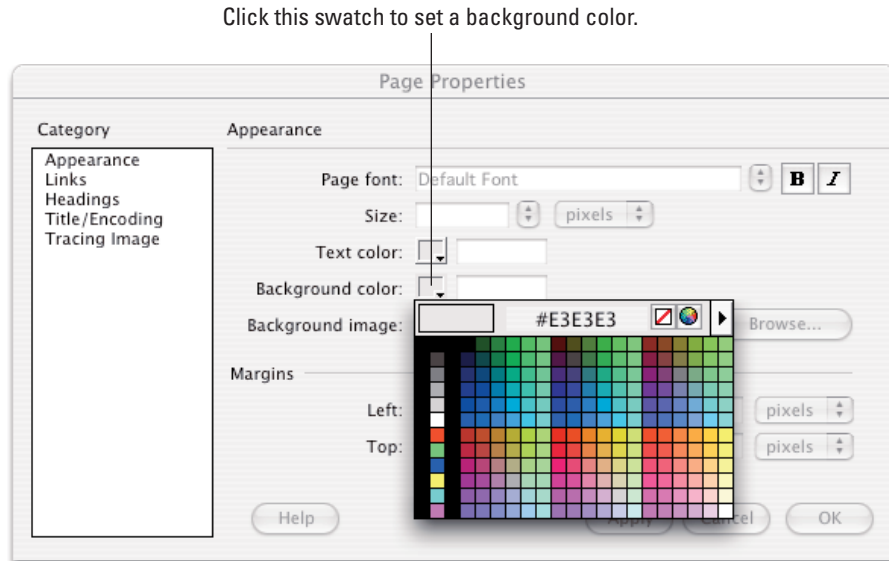
Figure 11-3: You should set Dreamweaver's background color to match your layout's color.

4. **Insert a text field**.

   On the main toolbar, change from the Common toolset to the Forms toolset by using the drop-down menu on the left side of toolbar. From the Forms toolbar in Figure 11-4, click the Text Field icon once to insert a field in your page. In the dialog boxes that appear, click OK to accept the default accessibility attributes and OK to add a form tag. In the document, you see a new field appear.

Text Field icon
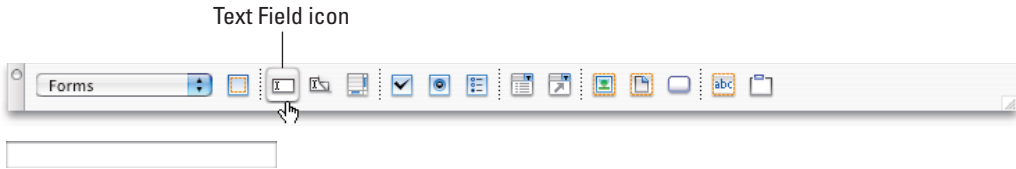
Figure 11-4: To insert form elements, first switch to the Forms toolbar and then click one of the Form options.

5. Insert a new form element.

Building a Web page in Dreamweaver is a lot like using a word processing tool. In the main document window, click once after the text field. You now see a blinking cursor. Press Enter to start a new line where you can add a new form element.

Back on the Forms toolbar, click the Insert Button icon. Click OK in the dialog boxes that appear, and in your document, a new button appears with the default Submit text, as shown in Figure 11-5.

Insert Button icon
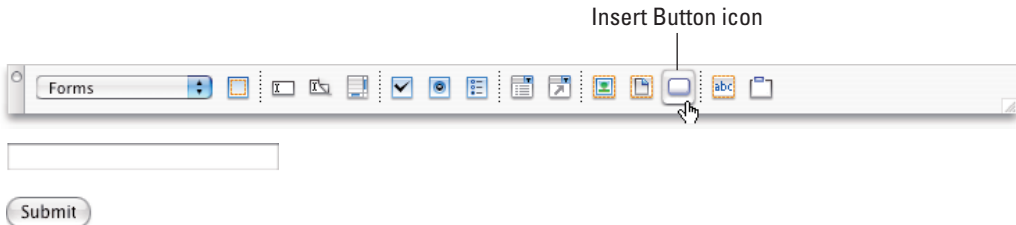
Submit

Figure 11-5: A Text field and Submit button ready to customize for your layout.

6. Customize the form graphics.

By default, the Properties inspector should be open. This is the long skinny palette shown in Figure 11-6 that allows you to adjust each of the elements in your document. If you do not see it, you can access it by choosing Window➪Properties.

In the document, click the text field to select it. In the Properties inspector, type **10** next to the Char Width box and press Enter to shrink the size of the field to accommodate just ten letters and numbers. By the way, the Properties inspector also lets you turn this text field into a longer scrolling multi-line field. To do so, choose the Multi Line option.

Next, select the Submit button in your document. In the Properties inspector, change the text in the Label box from Submit to Sign In or some other text appropriate for your layout.
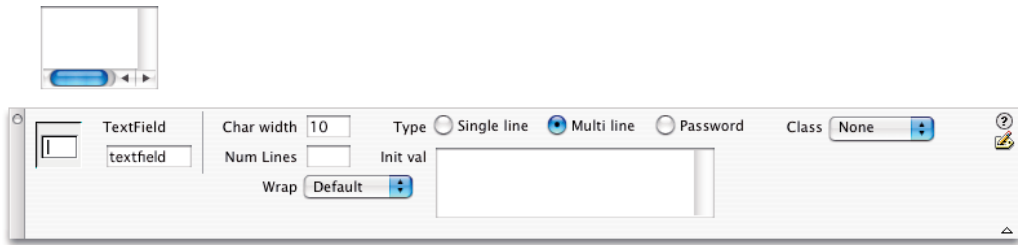
Figure 11-6: Customize your form elements by using the Properties inspector.

7. **Take a screenshot.**

   In your document, click once to the right of your new Sign In button and then press the Enter key. This deselects the form elements so that they don't have any funny selection lines around them.

   To take a screenshot, Mac users can press ⌘+Shift+4. After you do, the cursor turns into crosshairs. Click and drag a box around the form elements to take their picture. When you let go of the mouse button, you hear a camera shutter sound. The screenshot is on your hard drive or desktop and named `Picture 1`.

   Windows users press Print Scrn. A picture is taken of the screen and stored in the Clipboard (in memory). You must then immediately paste it into a graphics program and crop to just the element you need.

8. **Incorporate the screenshot in your Web page layout.**

   Open any graphics program, such as Fireworks, Paint Shop Pro, or Photoshop, and start a new file by choosing File⇨New from the menu. (In most graphics programs, a dialog box appears with the new file's options. For this example, create a file that is 600 x 400 pixels and 72 dpi.) Here's how to integrate the screenshot of the buttons into your new file:

   • **Mac users** choose File⇨Open and look for `Picture 1` on your hard drive (screenshots are always named sequentially as `Picture 1`, `Picture 2`, and so on).

   • **Windows users** have the screenshot stored in the Clipboard. Simply choose Edit⇨Paste to place the screenshot in your layout. The Windows Clipboard stores only one image at a time, so you must switch back and forth between the Web authoring software and the Web graphics software, capturing a screenshot and then pasting it.

TIP

After you open or paste the form graphic into your mock-up, sometimes you need to get rid of the white or colored background around it. Both Fireworks and Photoshop have a Magic Wand tool that you can use to select and delete the background color. To use it in either program, select it and before using,

adjust its tolerance to 0 (the default is usually 32) and uncheck the anti-alias option. Tolerance selects the amount of color variance to select when you click a single color. Zero tolerance selects just the single color. After adjusting the settings, click the unwanted background color around the form to select it and press the Delete or Backspace key to erase it.

Assembling an Online Presentation

With all your design directions in hand, your next creative task is to figure out the best way to present them to the client. I find that the best strategy is to develop both an online and an offline presentation. The online presentation gives the client the chance to see, in the browser, how the various designs will look on the computer. The offline presentation is a series of printed pages, mounted on nice boards for a polished effect. The printed portion of your presentation allows the client to quickly compare different designs side by side. Overall, the online/offline combination is a powerful one-two punch presentation.

This section focuses on how to create online presentations. Offline presentations are discussed later in the chapter.

Presenting your designs online

The best way to showcase all the design directions online is to build a special project Web site just for the client. Ideally, this Web site is password-protected so no one but you and the client can enter; however, if you're limited on resources, a *hidden Web address* should suffice. A hidden Web address is one so arcane that only you and the client could ever find it without knowing the direct path. An address like `www.yourcompany.com/workinprogress/client/round1` should do the trick.

TIP

The project Web site is a design repository for everything from the approved site map to the design directions. As you go, you'll find that you want to present more than one round of design directions, so the project site is also a good place to store past rounds so that a client can see the design's evolution.

Organize a project index page that provides links to the pages of each design direction. Figure 11-7 shows one such page created by David Solhaug of `www.sgrafik.com`. In addition, I find it useful to give a descriptive name to each design direction, such as `Direction 1: colorful, geometric`, just to make it easier to refer to them. I also like to provide a short paragraph that explains the logic and the benefits of each direction. Because this is the client's project site, they can refer to it on their own and share the address with coworkers. Without any guidance from you, they may not understand what they are looking at.
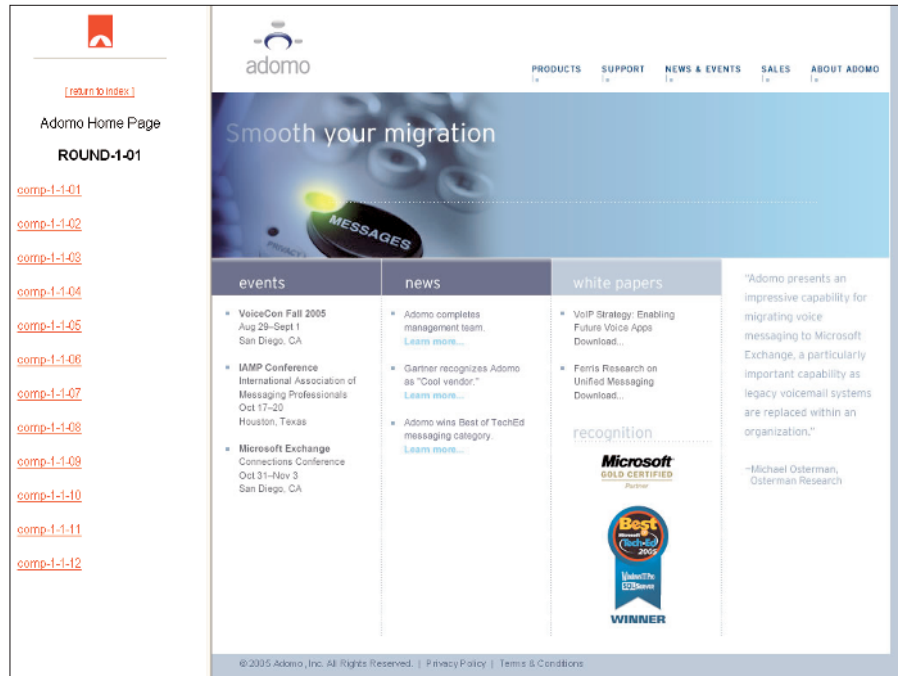
Figure 11-7: This site offers the client access to all current and past design directions.

The online presentation not only allows you to efficiently organize all the design directions in one convenient location, making it easy for a client to see all the directions, but also allows you a chance to include your own branding for that extra professional touch.

Constructing working and non-working prototypes

For the most part, you don't need to provide working HTML prototypes. Simple JPEG images should suffice to give a client a good idea of how the site will function. If you need to show how a complex interaction will work, you may consider showing a series of mock-ups in storyboard form rather than spending the extra time to make it really work.

In some cases, it may be worthwhile to make a small portion of the page work. In such cases, you can fake it by using HTML and various technologies that are easier to implement than the real deal. For example, you may find that in the bidding stage of a project, you may have to invest time in a semi-working, animated presentation simply for the wow factor, to help win the project in the first place.

Remote presentations

The Internet allows you as a Web designer to work with clients all over the world right from your home office. In fact, I never met the good folks at National Geographic in person, even though I designed at least three separate sites for them. All our presentations were carried out over the phone along with online presentations.

One of my favorite new tools for sharing presentations is Macromedia's Breeze (www.macromedia.com/software/breeze/). I simply circulate a URL to the clients, they log in to the Breeze meeting, and they can then see my screen as I move the cursor around to point out various elements. We are all connected on the phone via a conference bridge.

Creating Sizzling Printed Presentations

In addition to an online presentation (described earlier in the "Assembling an Online Presentation" section), you should definitely have a color printed presentation — if only for backup purposes. Just imagine not having access to the Internet during your client meeting. Aside from having a good backup, it's always a good idea to surround the customer with visuals. For one thing, unless you have access to an oversized monitor or projection system, it's hard to have the same impact with a small monitor when presenting to even a small group of people. Besides, on the computer screen, the client can see only one screen at a time. Comparing two design options side by side is impossible unless you have printed copies.

TIP

Creating black-and-white copies of the designs for each client present at the meeting is also a good idea. This way, they can take notes during the presentation and mark up ideas right on the design. Along with the black and white prints, include a copy of the site map and wireframes so clients can refer to them in the meeting.

In living color: Printing your mock-ups

These days, printing beautiful photographic-quality prints of your mock-ups is a simple and affordable process. The price of desktop color printers has come down remarkably in the last few years while the quality has increased ten-fold. For about $300, you can buy a great desktop color inkjet printer that gives you more than enough quality.

REMEMBER

When printing your mock-ups for a presentation, always print at the actual size and keep the resolution at 72 dpi. If you alter the resolution, shrink, or enlarge your mock-ups, you run the risk of making them appear blurred in the printout or too small for presentation purposes. In addition, print all your

designs in the same orientation — either landscape or vertical (I've found landscape works better), as shown in Figure 11-8. Standard 8½-x-11-inch paper is fine. Printing all in the same orientation on the same size paper helps to polish up the professionalism of your presentation.

TIP

Always use the more expensive photo grade glossy paper for your prints. If the paper isn't glossy, the ink soaks into the paper too much. When this happens, you can see the row lines of the ink dots — cheapening the effect of your presentation. Also, matte paper has a tendency to warp and curl when saturated with a lot of ink. When you print, look in the Print dialog box for the paper type settings and make sure to choose glossy paper, and Best quality for output settings. If you don't choose the correct paper in the print settings, the results aren't as good, and you waste a nice piece of glossy paper!

Figure 11-8: Most Web pages look better when printed and mounted in the landscape orientation.

Mounting your work on boards

Like a flimsy handshake or a wilting business card, your nicely colored prints just won't have maximum impact unless you mount them on sturdy boards. The standard in the Web design industry is to use black boards that are black all the way through. Don't use black-colored boards that have white interiors shining through along the cut edges. These have a tendency to fade even before you get to the presentation.

TIP

I like to use either black foam core or black Letramax brand boards. These two types of board, available at any art store, have ultra black finishes that don't detract from your designs. As for the thickness of the boards, the only thing that really matters is that the boards don't sag when you prop them up for display. Again, you must avoid the wilt factor! These two kinds of boards are good for mounting each design on separately. That way you have a series of boards for each home and subpage. If you want to paste all your designs on one board, try gator foam. It's a heavy duty board that comes in large sizes. You might need to shop around for it though. In the next section, I give you a few tips for mounting your prints on the boards.

Adhesive schmesive

You may think that I'm being a bit retentive, but the sticky stuff that you choose to mount your boards makes a big difference, not only in the final look but also in the flexibility you have for remounting. Rule number one: Never use glue! After you use glue, the boards and the print are ruined — you can never separate them from one another and reuse the boards. In addition, the drops of glue leave discernible lumps under your print.

The best adhesives to use are either 3M Spray Mount or a lightly adhesive, ultra-thin, double-sided tape. These allow you to remove your prints for remounting (in case you mess up) or to reuse the boards later. Spray Mount comes in an aerosol spray can and is pretty easy to use. Always use it in a well-ventilated area and use it *sparingly*.

WARNING!

If you use Spray Mount, don't spray the board! The spray goes everywhere and makes everything sticky. Instead, spray your print face-down with an oversized sheet of paper underneath to catch all the extra spray. Hold the can one to two feet away and make *one* pass. Let it dry for a minute or two and then mount the print to your board. If you use double-sided tape, place one strip at the top and one at the bottom of your print and then mount it to the board.

Consistency

One of the most important details in your presentation is to make sure everything is consistent. Use the same boards, use the same glossy paper, and make sure everything is cut to the same size! As shown in Figure 11-9, you don't want one board to be 11 x 14 while the rest are 14 x 17. Consistency ensures that the focus is on the designs and not the irregularities in the presentation.

Here are two more tips to follow when preparing your printed presentation:

- **Leave breathing space around each image.** When you print your mock-ups on 8½-x-11-inch glossy paper, you may be tempted to trim away the white borders, leaving just the printed design. I find, however, that trimming your designs so tightly is like cutting your fingernails too close. I recommend leaving a little breathing room around your images. Either leave the prints centered on the 8½ x 11 page, or trim them all to the same size with at least a one-inch border all around.

- **Cut the boards to a manageable size.** Aside from cutting all the boards to the same size, make sure they are neither too big nor too small for the prints. If your prints are 8½ x 11, a good size for the boards is 11 x 14. The largest you should go is about 14 x 17 for individual boards each holding one design.

Figure 11-9: The one board that has a different size and a vertical print stands out unfairly from the rest.

Presenting to Clients

Now, all that's left is to sell the client on your ideas and remain professional, confident, and composed during the feedback. There's no special sauce to successfully pitching a client (other than a good cup of coffee), but I can offer some behavioral guidelines:

- **Dress the part.** On presentation day, make a point to dress appropriately. Even if you are presenting to a bunch of engineers in jeans and tennis shoes, you look more professional if you have some style (in other words, brush your hair and take a shower).

✔ **Be the discussion moderator.** Lead the show by presenting each design direction objectively as if you weren't the designer. Openly discuss your own professional opinions about what works about each direction and why. Ask for the clients' opinions and be genuinely open to their suggestions and concerns.

✔ **Don't be married to your work.** Although you may have a favorite design, don't be defensive about any one design. You must be the champion for your clients and help them pick a design that works for them. The best way to sway clients toward your favorite design is to talk objectively about the design's benefits. If clients express objections about the design, don't take it personally. Either concede and acknowledge their point, or find a logical counterpoint for them to consider.

✔ **Be confident, be positive; never berate your own work.** Always remember that design is a subjective topic, and no one ever likes the same things. But don't anticipate rejection by undermining your own work. Remember, if it's good enough to show, it's good enough to stand by.

Clients are suckers for the "ugly duckling"

You can count on at least one thing during a presentation: The client will pick the ugliest design in the group. A word to the wise, therefore, is to be careful about which design directions you decide to include in the presentation. Make sure that you can live with any idea a client may pick.

Another thing to remember is that some clients have a knack for taking things too literally. If they see something they don't like or something that doesn't make sense, they might reject the entire design. For example, if they see a bad photograph that's just a stand-in for a future photo, they may dismiss the entire design. The same holds true for headlines and dummy text. If the headline doesn't make sense, they focus on it and worry that you are suggesting final copy. The solution is to use greek text for most text areas, and at the start of the presentation, make any necessary disclaimers about legible copy and images.

Here we go again: Round two

More often than not, your designs won't hit the nail on the head the first time around. Be prepared for the client to like certain aspects of each design and ask you to do another round that combines the various elements into one new design. I call this second design round the Frankenstein round.

This is actually a healthy process, believe it or not, because you're still the designer and can control how the various aspects are combined into one. Plus, at the end of this round, you should be that much closer to something the client loves.

WARNING!

The worst scenario, however, is to let clients try their hand at assembling the Frankenstein round themselves. This can truly result in a monster of unbridled proportions. When clients gets their hands on it, their heart and their ego get involved too. The resulting design is likely to be a horrific mess that needs a lot of help. Unlike you, with your professional distance, a client might not take kindly to honest criticism, and there you are, stuck with their mess.

The other "gotcha" is to let the client request too many design changes. I have worked for large corporations that noodle a poor vendor's designs over and over. Not only does this make timelines slip, it also makes profit margins slip. Be upfront in your initial proposals about the number of design revisions included in your bid. Tell clients that any noodling past the second round incurs hourly rate charges above and beyond the fixed bid amount.

12

Dome. Bring your own g
We'll provide food and a d
hour video on scaling the

Polishing Pixels to Perfection: Graphic Production

*A*fter the client approves the initial design direction, the bulk of your Web design time is spent glued to the monitor pushing pixels around with the mouse, which is akin to drawing with a bar of soap. Needless to say, graphic production takes up the lion's share of time spent in Web design (aside from the technical integration stuff) and is an art form — and a tedium — in and of itself.

The object of Web graphic production is to maximize the quality of the graphic while minimizing its down-load time — two goals that are diametric opposites. The higher the quality, the larger the file size, and thus, the slower the download. In this chapter, you find out how to get the most out of your graphics by knowing which file formats to choose, and how to maximize download performance by reusing the same graphics on every page. I also show you how to slice images into sections that you can save with different file formats to maximize quality, how to prepare images to match the background color or tile of a Web page, the significance of aliased and anti-aliased graphics, and how to prepare transparent graphics.

To streamline the production process, you should design templates for each different kind of graphical element — from navigation banners and headings to buttons and images. This way, you can delegate the final production work

to a whole team while ensuring a consistent look and feel throughout the site. In this chapter, you find the ins and outs of Web graphic production so you can be polishing pixels to perfection in no time.

Graphic Production

Pumping out all the graphics that you need for a Web site is a daunting task. You must prepare hundreds of individual graphics. The best way to tackle this effort is to make templates that you and your production team can use for every different kind of graphic and layout. For graphics that you can't make templates for, create a style guide that covers issues on font choices, color choices, and so on. With the help of templates and style guides, you can delegate a lot of the production so that no one person is the "keeper of the kingdom."

REMEMBER When only one person knows how to design the site, not only do you create a production bottleneck, but clients can never update their site without your help. This sounds like a brilliant plot to retain a client's business, but I think this is bad form. The client came to you in the first place for your design help, so providing other team members, and ultimately the client, with the tools, templates, and style guides is key.

Design templates

TIP When you make templates for the team, you should adopt one graphics program such as Fireworks or Photoshop that the team can use. Provide templates in the native format of one of these programs. This way, the team members can open a template, edit a design piece, optimize it, and export it ready for the Web site.

You can make graphical templates for almost everything on the Web site — including entire page layouts. If you recall, most Web sites have just a few basic layouts — the home page, a subpage, and perhaps a few different variations of the subpage's interior to keep it interesting or to accommodate special content like forms and tables.

Here are some of the different kinds of templates that you can prepare for a Web site:

- **Home page layout.** By providing a complete mock-up of the home page, you give the HTML people who assemble the actual page a visual guide to go by. In addition, a complete mock-up is your chance to show how all the different components — headings, buttons, and bullets — work together on the page.

✔ **Subpage layout.** Like the home page mock-up, a complete subpage design acts as a visual guide for both the HTML people and the graphic production folks. A complete mock-up gives context to the design elements, and, if push comes to shove, production folks can use graphics from this template to build interface elements for the site.

✔ **Variations on the subpage.** Because most pages of a Web site use the subpage template, it's a good idea to provide a few different layout variations. For the most part, the variations should focus on the interior of the page — always leaving the main navigation system in place.

✔ **Global navigation.** Although the navigation system is probably included on the home and subpage templates, it's a good idea to save it as a separate template. As a separate template, it's easy to update.

✔ **Graphical buttons.** A Web site often contains a lot of graphical buttons, such as Submit, Next, Previous, and so on. Provide a template that has the basic button design — its color, shape, shadow, and bevel. Then, a production team member can open the template, make changes to the text only, and export buttons that have a consistent look.

✔ **Graphical headings and subheadings.** As you did for the button template, provide a template for all graphical headings and subheadings set in the right font, size, and color. This way, a production team member can open the template and simply change the content of the text without worrying about its formatting.

✔ **Graphical bullets.** Again, to ensure consistency throughout the site, provide a set of graphical bullets, arrows, and other widgets. For example, a team member can open the template, change the color of a bullet, and export it.

✔ **Image treatments.** If you have designed a special treatment for images on particular pages, you should provide a template with the proper sizing, masks, borders, and coloration. For example, if all home page images are steel-gray monotone with a soft feathered edge on one side, you can set up a series of layer treatments in Photoshop that can do this automatically to any new image. This way, you can easily produce a series of images that all have the same graphical treatment.

✔ **Background tiles.** If your Web site uses background tiles, always keep the original art on hand in a template in case the team needs to make changes.

Style guides

If the site that you're designing is a large one, you probably need a whole team of HTML people to create it. They must all use the same font, size, and color treatment for the headings, subheadings, links, and other elements of a page.

Accommodating other languages

If you're producing graphic templates for a site that will ultimately be translated into other languages, all areas with graphical text must have enough room in the design to expand 30 percent in all directions. For example, if you have a navigation bar with graphical text, don't make the bar too narrow. Chinese characters are much taller than English characters. Similarly, German words are generally longer than English words, so ensure your designs allow for horizontal expansion. See the navigation scheme in the following figure. When the English is translated into Chinese and German, the design breaks. ION Global, `www.ionglobal.com`, are experts in site *localization* as it is called in the industry and share some interesting case studies on its site.

The IFC site is available in both Chinese and English. The visual design allows for 30 percent vertical and horizontal expansion of the text elements both in the navigation and in the body.

Example provided by IONGlobal.com. http://ifc.com.hk

TIP

You can either provide the style guide as a printed document or, better yet, as a CSS file (Cascading Style Sheets). As a CSS document, it's ready for use in the site. Either way, provide a style guide that covers the following:

- ✔ **Headings and subheadings.** Provide font, color, and size guidelines.

- ✔ **Body text.** Provide font, color, and size guidelines.

- ✔ **Captions.** If different kinds of captions are needed for different kinds of elements, the style guide should address how and when to use each caption type.

- ✔ **Pull quotes.** If a Web page ever needs a quote or other statement to stand out from the main body text, the style guide should address what font and size to use.

- ✔ **Links.** In addition to font and size guidelines, choose a color for the link in all its states. (Links often display in one color before they have been clicked and another afterwards.)

- ✔ **Table treatments.** If you use tables in a Web site, decide whether the tables should have border colors or background colors in each cell. Also, you may specify different font, color, and size treatments for the text within tables.

- ✔ **Background colors and tiles.** Finally, if certain areas of the Web site use different background colors, provide a list of the colors to use. For example, the Products section of the Web site may use a different background color from the About the Company section. The same holds true for tiles. Some pages or sections may use tiles, and others may not. The style guide should provide a road map for the use of background colors and tiles throughout the site.

Version control

For large sites and large teams, keeping track of the latest version of each graphical template, or *comp,* as they are called, can be an issue. Many people could be working on the files, sometimes the same file at the same time. The solution in such a scenario is to use a document management system such as Microsoft's Visual Source Safe, or *VSS.*

With a program like this, you can check in files to a shared server. When team members want to work on it, they check it out, which prevents others from working on it. When the file is checked back in, VSS keeps a record of each new version and makes it possible to open a previous version if the new one is ruined for any reason. For smaller teams and sites, try using a naming convention that appends the date and time at the end of the filename.

Fun with File Formats

From the template comps, team members can begin to export all the little graphical pieces needed for the site, from buttons to headers. You must optimize these exported pieces for Web delivery, yet still look great. The key to achieving this balance is an understanding of file formats.

Although you can choose from a few different kinds of Web graphic formats, for the most part, you use the GIF or JPEG format for your images. Which one you choose depends entirely upon the type of image that you're dealing with.

When to use GIF

The *Graphics Interchange Format* (GIF) is one of the older file formats on the Web. GIF has evolved over the years to include a number of features that make it a handy file format for a lot of different types of graphics — including animation.

TIP

GIF is best suited for graphics that have non-photographic elements in them, such as text and flat-colored graphical areas. This is because GIF is a *lossless* compression format (see the nearby sidebar). Unlike the *Joint Photographic Experts Group* format (JPEG), GIF doesn't corrode the quality of your images.

GIF and JPEG image compression

The two most common graphic file formats on the Web, GIF and JPEG, were designed to compress an image's file size as much as possible. (By file size, I mean the number of kilobytes it occupies after you save it.) These two file formats use algorithms (mathematical formulas) to figure out the best way to save the image using the least amount of disk space — a process called *compression.*

GIF: GIF uses an *RLE (Run Length Encoding)* compression scheme that looks across each horizontal row of pixels and records changes in pixel color by replacing areas of same-colored pixels with a short code, thereby reducing file size. If your image has large areas of solid color (like a cartoon, for example), it compresses better than

a photograph with subtle tone changes. For this reason, use GIF for solid-colored images, but not photos. GIF does not alter the actual pixels in an image. For this reason, it's considered a *lossless* format.

JPEG: The JPEG compression algorithm, on the other hand, works by modifying and averaging the color data of small blocks within the image. This method degrades the quality of an image and introduces noise called *artifacts.* JPEG, therefore, is considered a *lossy* format. The artifacts aren't as noticeable in photographs as they are in flat-colored graphics such as graphic text and illustrations. For this reason, JPEG works great for photos, but not so well for flat-colored graphics.

Figure 12-1 shows what text looks like when it's saved in GIF versus JPEG format. Zoomed in, the GIF image is crisp and lossless — no detail has been lost, even though I reduced the palette to only six colors, making the image 2K. I saved the example on the right in JPEG format at 50% quality, making it 4K — double the size and half the quality. Notice the severe amount of degradation in and around the text. This sort of degradation is less noticeable in photographs, which is why the JPEG format works so well for photographic images.

The GIF format leaves crisp edges.

JPEG compression can leave substantial artifacts.

The original file

Figure 12-1: The quality of an image saved in GIF compared to a JPEG.

The original image in Figure 12-1 contains only flat-colored areas and text, so it's a no-brainer to choose GIF. A problem arises, however, when a graphic contains text, flat-colored areas, *and* photographs. Which should you choose: GIF or JPEG? You can answer that question in two ways:

  ✔ **Use GIF with an *adaptive palette*.** The adaptive palette chooses the best colors for an image. (See Chapter 9 for more information on the adaptive palette.) This way, you can retain the quality and readability of the text and graphics areas while still getting the best color fidelity for

the photograph. The caveat, of course, is that in order to get the best quality, you need to use a lot of colors in the palette, which increases the file size.

WEB SPEAK

✔ **Chop, or *slice,* as the industry likes to say, the graphic into pieces.** You can see an example in Figure 12-2. This way, you can save the text and graphics portion as a GIF with few colors in the palette, and save the image portion as a JPEG. For more on slicing, see "Slicing and dicing Web graphics," later in the chapter.

Text exported as GIF

Colorful texture plus text exported as JPEG
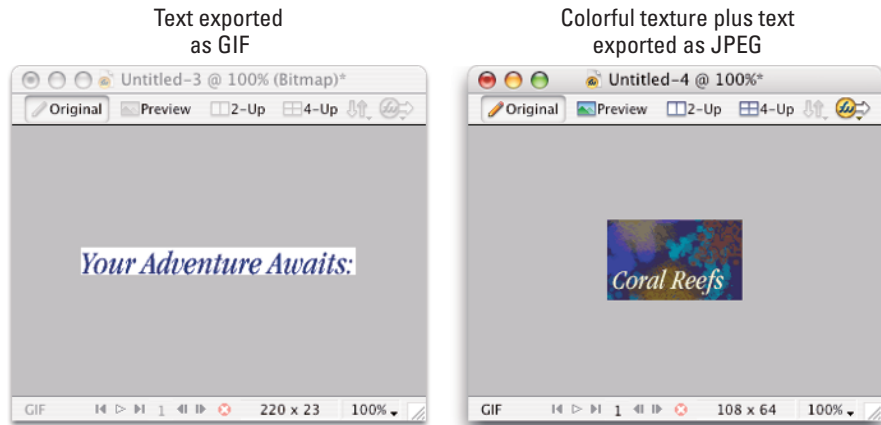
Figure 12-2: By slicing a graphic in two, you can save each portion individually.

GIF transparency

Another feature of GIF that makes it an interesting choice is its ability to make a color transparent. This feature comes in handy when you want to place an image on top of a pattern, such as a background tile. By making the background color around your image transparent, the image appears to float on top of the background pattern. If you don't make the background color transparent, you end up with a big block sitting on top of the pattern in your Web page.

REMEMBER

When you create a graphic, it *always* has rectangular dimensions. During the creative process, you can choose a background color upon which to build your graphic. When you export the graphic as a GIF, the background color that you chose goes with it. If you don't choose a background color and work on a transparent layer in Photoshop or Fireworks, the background defaults to

white when you export. Either way, you end up with a file that has rectangular dimensions — the image sitting on a white or colored rectangular background, as shown in Figure 12-3.

Photoshop's transparency works
only in Photoshop.

After you export, the background
defaults to white.

Figure 12-3: The transparency in this case is native only to Photoshop. Once exported, the background is a solid white.

The only way to get rid of this background is to make it transparent. When you export a graphic as a GIF, you have the option to choose a single color as the transparent color. If you choose the background color surrounding your image as the transparent color, your image remains visible and appears to float on top of the patterned background tile of the Web page, as shown in Figure 12-4.
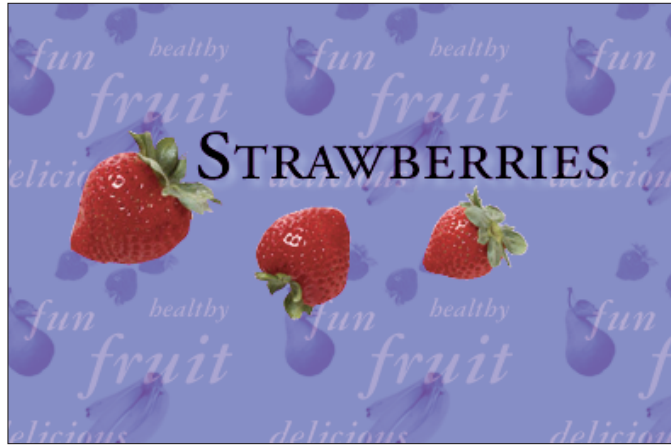
Figure 12-4: If you make the background transparent, the image floats on a patterned page.

Before you get too excited about the transparency feature, however, you need to be aware of a few "gotchas":

WARNING!

✔ When you designate a color as transparent, make sure the color isn't part of your image. Otherwise, your image can end up looking like Swiss cheese. For example, if you build your graphic on top of a white background and choose white as the transparent color, *all* white areas become transparent. Imagine the whites of people's eyes becoming transparent! Take a look at the ghostly image in Figure 12-5.

✔ If you want to use a patterned background tile in your Web page, you must use transparent GIFs in order to achieve the floating effect. Don't think you can simply build your graphic on top of a copy of the pattern and hope that it matches up with the pattern on your page — it won't. Each browser begins the tiled pattern in a different spot, so you never know where the pattern falls on the page. You end up with an obvious mis-alignment of patterns, as illustrated in Figure 12-6.

Figure 12-5: Parts of this image vanish because they are the same color as the designated transparent background.

Figure 12-6: Matching up a background tile with the pattern in an image is nearly impossible.

TIP

✔ The transparency feature comes in handy — even if you aren't using a background pattern in your Web page. If you are using a solid background color in your Web page, always build your graphics on a matching background color. Then, when you export, *make that color transparent!*

Although this technique sounds silly, it actually prevents a very common problem. Remember that different monitors have different bit depths and display even Web-safe colors a little differently. Even if the block of color surrounding your graphic matches the background color of the Web page, you can see a slight difference on some monitors, as shown in Figure 12-7. Without using transparency, some people can see the subtle outline of the square shape of your image.

Background color of graphic

Background color of Web page
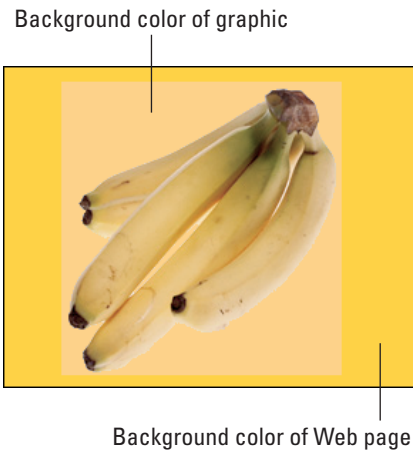
Figure 12-7: Some monitors display background colors from your file and from your Web page differently.

GIF animation

Not only is GIF a great all-around format for saving Web graphics with transparency, but it's also great for building simple flip book animations.

To create an animated GIF, create a series of individual GIF files that are all the same size and then string them together. The easiest way to do this is to use a Web graphics program such as Fireworks. The individual graphics are

all saved together in one animated GIF file. You even have control over the timing (how long each GIF appears before flipping to the next one). You can make the animation play continuously in a loop, or you can specify a certain number of loops — from just one time through, to five or so repeats.

A drawback to GIF animations is that you can't use a lot of frames. Because each frame is a separate GIF image, a lot of frames can add up to a beefy file that takes forever to download. With fewer frames, the animation downloads and plays faster, but the action looks choppy. GIF animations work best for cycling messages, but not for complicated creations like animated characters, which must have a lot of frames to play smoothly. For that kind of animation, Macromedia Flash yields way better results.

Take a look at the animated GIF banner in Figure 12-8. The animation has only four frames that cycle through a few different messages. The whole thing is just 10K — an ideal file size for a banner ad. Also notice that the graphics use solid-colored areas and text, making it possible to use just six colors for the whole enchilada.

Figure 12-8: This animated GIF banner has four frames that emphasize words in a message and end in a call to action.

Dithering: Pixel pointillism

You can't save an image as a GIF until you reduce its palette of millions of colors down to 256 or fewer colors. (Only having 256 colors is a limitation of the GIF format. For more information, see Chapter 9.) When you reduce an image's palette, it becomes *dithered*.

Dithering is the process of placing two or more colors in close proximity to visually simulate a missing color. For example, if you want to draw a picture of an orange, but you have only red and yellow markers, you have to do some creative pointillism to approximate the missing orange color. The same thing happens when you remove millions of colors from your image. The limited palette of 256 colors must work in concert to simulate the missing millions of colors.

Although dithering does degrade the quality of your images, you can compensate somewhat by using an adaptive palette when you convert your image. An adaptive palette chooses the best colors for the image. This helps to keep the dithering to a minimum, often resulting in an image that looks as good as the original. You are free to export the image as a GIF after reducing the image to 256 or fewer colors.

When to use JPEGs

The JPEG format was specifically designed to compress photographs for Web delivery. Although GIF is limited to images with 256 or fewer colors, the JPEG format works only for 24-bit images that can contain millions of colors — ideal for the gradients and subtle blends common in photographs.

WEB SPEAK

Images that are saved as JPEGs don't dither because the image is free to use a ton of colors. When you save an image as a JPEG, however, the format introduces some *noise* (also called *artifacts*) in the image — especially as you increase the amount of compression. Most programs like Photoshop and Fireworks allow you to adjust the quality percentage of a JPEG image. The lesser the quality percentage, the smaller the file size, but the greater the noise. I find that 60 to 80% quality is a good range.

In a photograph, you don't notice the noise, but if you save an image with flat-colored graphics and text, the artifacts really stand out as shown in Figure 12-9. This is why the JPEG format is considered a *lossy* format: It corrodes the integrity of your original image.

TIP

Use JPEGs on purely photographic images. Use GIF for flat-colored graphics with text. For images that have a mix of photography, flat-colored graphics, and text, try both a high quality (like 80%) JPEG and an adaptive palette, 128 color GIF and see which looks better and produces a lower file size.

Figure 12-9: The photo on the left looks fine, but the image on the right shows a lot of artifacts — a byproduct of JPEG compression especially evident on text elements.

Lean, Mean Page Design

Your goal as a Web designer is to create a nice-looking, user-friendly page that downloads in a reasonable amount of time. After all, a big part of the user experience is NOT waiting for a page to download. The longer it takes, the worse the user's experience — no matter how nicely you design the site.

The best way to speed up the download time for a Web page is to use graphics efficiently throughout your Web site. Take advantage of design features, such as transparency, background tiles, and the browser's ability to store graphics in its memory to maximize each page's performance. In addition, by chopping graphics into individual pieces, you can apply the most efficient file format and palette settings to each one. Not only does this give you more control over the quality of each individual graphic, but it also helps you to shave precious kilobytes off your Web page, making for a speedier delivery.

Minimizing download times

In addition to choosing the right file format for an image, whether it's a GIF or a JPEG, you can use other strategies to reduce the file size of your graphics and speed up a page's download time.

Here are a few techniques to add to your arsenal to get the most bang for your graphics buck:

- **Use transparent colors.** By simply choosing a transparent color for a graphic, you can reduce its file size. The transparent color becomes one less color for the browser to worry about.

- **Repeat background tiles.** Rather than including huge graphics on your Web page to add visual interest, get creative with repeating background

tiles. Figure 12-10 shows a rather complex Web page designed by Juxt Interactive as a demonstration for Macromedia. The page looks graphically intense as a result of a clever and fast-loading background tile, shown in Figure 12-11, that's reused on every page. Although the tile is large, the file size is only 15K because the image uses so few colors. In addition, because it is reused on each page, the browser does not have to download it again on subsequent pages — it simply redraws it from memory.

By the way, this tile is much larger than the Web page because of the repeating nature of tiles. In this case, the designers don't want the tile to repeat until well beyond the visual bounds of the browser window. Users would have to scroll to see this pattern repeat again.

✔ **Reuse graphics.** Whenever you can reuse graphics on a Web page, do so: It speeds performance. When a graphic downloads in a Web page, the browser *caches* it — stores it in its memory. Navigation graphics are classic examples of graphics reused on every page. These graphics, as shown in Figure 12-12, load quickly on all pages because the browser redraws them from memory. This process is a lot faster than downloading a new image.

Figure 12-10: Much of the rich graphical nature of this page is due to a creative background tile.
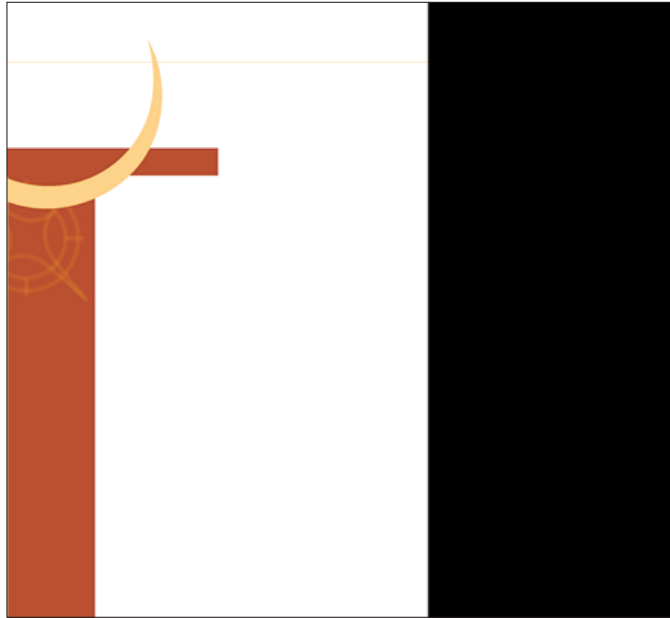
Figure 12-11: This background tile provides visual interest at a relatively small file size.
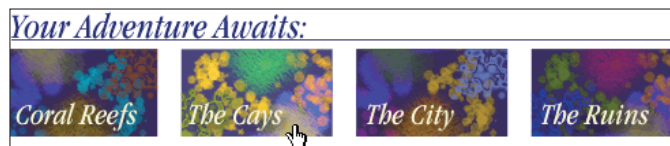
Figure 12-12: After this navigation bar downloads, the browser caches it.

Slicing and dicing Web graphics

Another way to optimize your page design is to chop, or *slice,* graphics into individual pieces. This technique accomplishes three things:

✔ Users don't have to wait for large blocks of graphics to download before they see anything on the page. The little individual pieces load relatively quickly, so users see at least part of the page design almost immediately.

WARNING!

Don't get carried away with your image-slicing. Loading hundreds of tiny graphics can actually slow the page down.

✔ You can save the sliced graphics individually. This means that you can apply different file format and palette settings to each one to get the best results. For example, as shown in Figure 12-2, the title area uses just one color, whereas the buttons are almost photographic. If you save the title separately, you can save it as a four-color GIF. If you keep it grouped with the buttons, however, you must choose a palette of at least 100 colors to accommodate both the text and the graphics.

✔ Sliced graphics can act as independent moving parts in your Web page. For example, you can turn an individual graphic into a *rollover button* — which changes its appearance as the user's mouse pointer rolls over it. Because the button is sliced as a separate graphic, none of the other graphics on the page are affected when the user's mouse pointer rolls over it, as shown in Figure 12-13.

Figure 12-13: Each button in this navigation bar is sliced as a separate graphic.

Most graphics programs such as Fireworks and Photoshop allow you to slice your Web graphics into independent parts. After you slice the graphics, these programs also enable you to add interactive links and rollovers to your graphics, assign links to them, and apply different file format and palette settings.

You get a lot of great benefits when you slice a graphic, but you may be wondering how in the heck to piece them all together again. The answer is to use an HTML table structure. Fortunately, both Fireworks and Photoshop can output the necessary HTML to make all your sliced and diced graphics come back together again correctly in the Web browser.

Backgrounds for Graphics

Before I turn you loose to churn out Web graphics, I want to equip your brain with one more production note. Above all else, the most common mistake you can make in producing Web graphics is to not properly prepare the edges of graphics so that they match the background.

WARNING!

If you don't proactively address the edges of your graphics, you can end up with a file that looks like it's sporting a halo, as shown in Figure 12-14. This image was built on a white background. Although the designer chose white as the transparent color, the semi-white edges around the image remain. Placed on a dark background, the white rim really stands out.
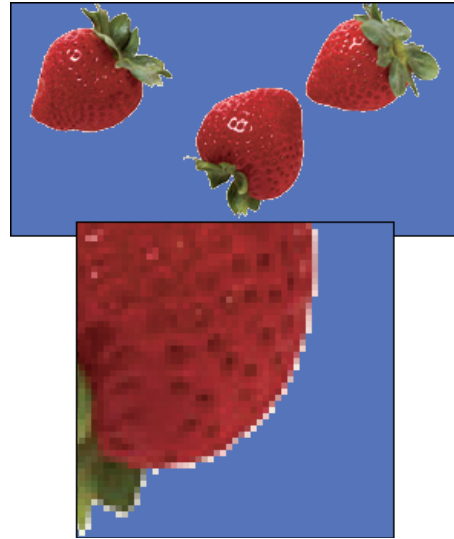
Figure 12-14: This image has a halo because its edges were not properly prepared for the dark background.

Understanding anti-aliased and aliased graphics

This white halo is the result of *anti-aliased* edges around the image. Because images are made out of pixels, and because pixels are square, an image with a curved edge looks jagged. To alleviate this problem and make the curved edges appear smooth, graphics programs include an outer rim of semitransparent pixels around the curves of images and text — a process called *anti-aliasing*. The semitransparent pixels blend the stair-stepped edge into the surrounding background color, as shown in Figure 12-14.

When you export graphics from a graphics program, you always end up with a solid background color (even if you were working on a transparent layer in Photoshop or Fireworks). In the exported graphic, the semitransparent edges also become solid-colored pixels. The color of these pixels is halfway between the background color and the image color.

This means that when you choose the background color as your *one* transparent color, the transparency stops just short of the blended in-between colors, leaving a rim around your image. You won't see the rim if the graphic is placed on the same background color it was anti-aliased to. You do see it, however, if the image is placed on any other background color.

TIP

The best way to avoid the halo is to always build your graphics on the same background color that you use on your Web page. This way, when you set the background color to be transparent, the anti-aliased pixels are pre-blended to match your Web page.

WARNING!

The one caveat to this method, however, is that by pre-blending the edges of your graphic, you lock yourself into using the graphic only on pages with the correct background color. For example, you can't reuse the graphic on a page with a different background color or the halo appears again. (So much for caching the graphic!)

The way to avoid this is to use graphics with *aliased* edges — ones with no intermediate pixels between the image and the background. Keep in mind, however, that graphics with aliased edges appear jagged. For example, HTML text is aliased text: That's why it appears so chunky. Aliased edges, however, are also the reason why HTML text looks fine on all background colors and patterns. Without a rim of blended pixels to worry about, aliased graphics can go anywhere.

Preparing graphics with drop shadows

The halo problem is even more pronounced when your image has a *drop shadow*. After all, a drop shadow is really just anti-aliasing on steroids. Take a look at Figure 12-15. Because I built this image on a transparent layer in Fireworks, it's easy to assume that I can export the image and have it retain the transparent effect — especially because I chose the transparency option when I exported the image. The problem, however, is that this transparency is native only to Fireworks for the purposes of building graphics. When you export, the image defaults to a white background, and the GIF transparency only affects the one solid white background color, leaving behind a big blob for the drop shadow.

Images on transparent layers have white backgrounds when exported.
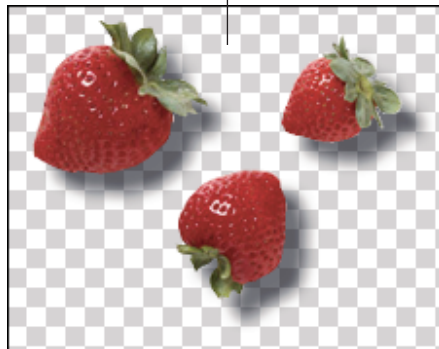
The drop shadow was blended to match the default white background, not the black Web page background color.

Figure 12-15: The halo problem becomes an eyesore when an image has a drop shadow.

To get some hands-on experience, in the next few steps I show you how to use Fireworks to prepare and export an image with a drop shadow ready for the Web:

1. **Launch Fireworks and start a new file.**

   Choose File⇨New from the menu bar. In the dialog box that appears, enter the dimensions 200 x 200 pixels, 72 dpi, so you have enough space to work. Next, choose the Custom color option and click the color swatch to choose a color that matches your HTML page (this way, your graphics will be anti-aliased to the proper color). For this example, just choose any old color and click OK.

2. **Create a circular shape.**

   In the Tools panel, select the Ellipse shape tool (click and hold the Rectangle tool to reveal the Ellipse tool option). Draw a simple circle in your document, as shown in Figure 12-16. When you let go, the circle should still be selected. With the circle selected, you can choose a new fill color for it by clicking the color swatch at the bottom of the Tools panel, as shown in Figure 12-16.
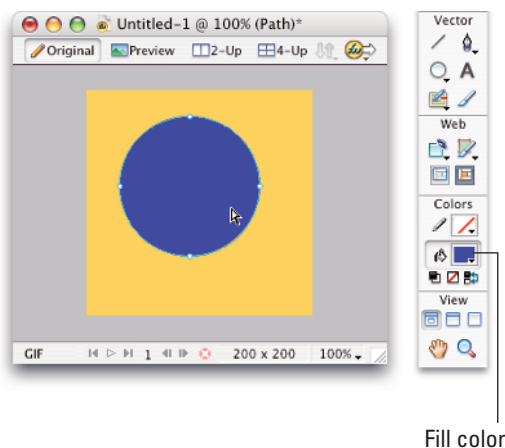
Fill color

Figure 12-16: Choose a new color for your selected circle by clicking on the Fill Color swatch.

For this example, you are drawing a circle to achieve curved edges that will be anti-aliased to the background color. Shapes with straight edges, like a square, will not be anti-aliased because they have no rounded corners to smooth out.

3. **Add a drop shadow.**

To add a soft drop shadow to the circle, locate the Effects option in the Property Inspector. As shown in Figure 12-17, next to Effects, click the + symbol and choose Shadow and Glow⇨Drop Shadow from the drop-down menu. After you select the effect, a mini window appears with setting options. Click anywhere outside this window to accept the default settings and close the window.
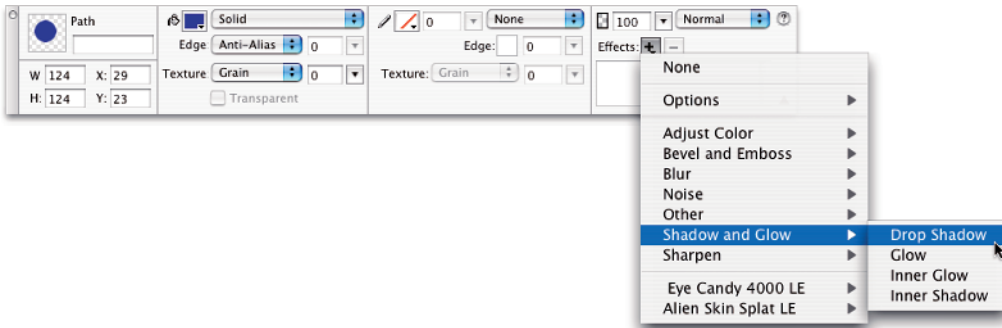
Figure 12-17: The Effects option enables you to apply drop shadows and bevels to any graphic or text object.

4. **Optimize the graphic as a transparent GIF.**

If it isn't open already, open the Optimize panel from the Window menu. In the Optimize panel, choose GIF Adaptive palette, 128 colors from the list of choices. Then choose Index Transparency to make the background color transparent. Click the color swatch next to Matte and move the Eyedropper over to and click your background color. In your document, click the Preview tab. The result should look similar to Figure 12-18.

To export this image for the Web, choose File⇨Export. In the Export dialog box, name your new image and then save it.

Matching a background tile

What if you want to match a graphic to a background pattern? The production process for both procedures is nearly identical. The only difference is that you must pick a median-valued color from the pattern and use that as your background color when preparing your graphic for export. When you set up the median color as the transparent color, the anti-aliased pixels that remain match the overall color of the pattern.

Click Preview tab   Set Matte color to match background
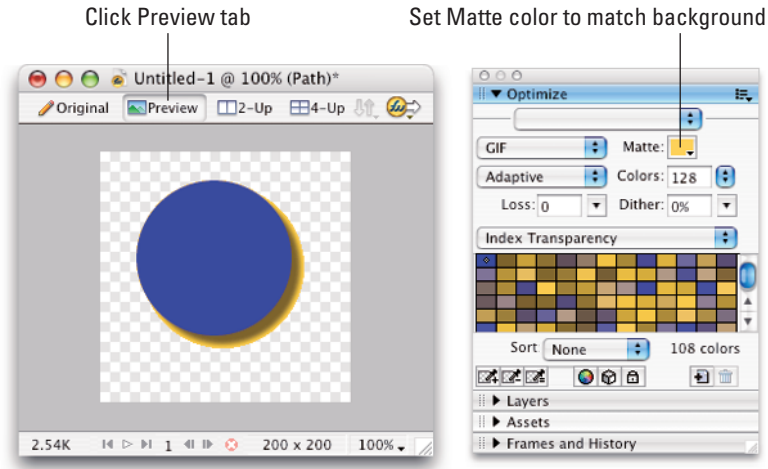
Figure 12-18: Choose GIF Adaptive palette, 128 colors, and then choose the Index Transparency option.

Using alpha channels

All digital images are comprised of three *channels* — red, green, and blue (the RGB system) — that create the image you see. Each image can have an additional invisible channel called an *alpha channel* whose sole purpose on earth is to control the transparency of the RGB image. (Photoshop uses multiple alpha channels to store selections for the purposes of image editing, but that's a whole other subject.)

To control transparency, an alpha channel uses a grayscale image (an image colored in shades of gray, white, and black). Like a digital stencil, completely black areas make the image transparent in those spots, whereas completely white areas let the image shine through in all its glory. Gray colors in the alpha channel equate to varying degrees of transparency — the darker the gray, the more transparent the image is.

GIF allows you to specify multiple transparent colors, but it doesn't let you specify degrees of transparency. The PNG format, however, does support true semitransparent alpha channel masking. At the time of this writing, however, I have yet to see a semitransparent PNG in action on a Web page. This is truly the panacea to creating anti-aliased graphics with drop shadows that can work on any background color, such as the one shown in Figure 12-19. The benefit of true alpha channel transparency is that you can

create a graphic once and reuse it throughout a Web site regardless of a page's background color or texture. This is in stark contrast to today's practice of creating multiple editions of the same graphic — one for each background color it uses.
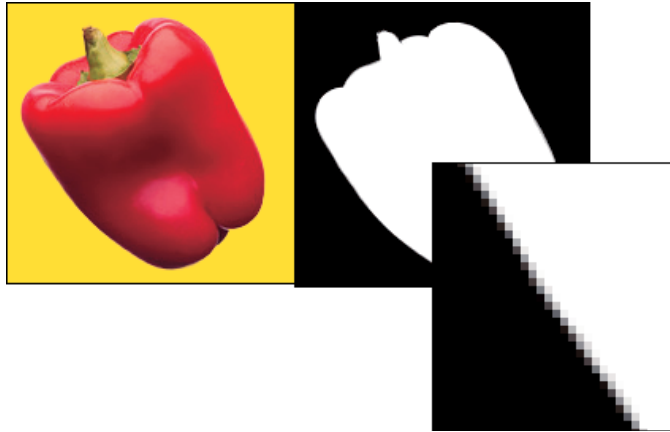
Figure 12-19: An alpha channel enables you to make the edges of an image semitransparent.

Part IV
Producing the Final Web Site

The 5th Wave — By Rich Tennant

MIDTOWN

WHERE'S THE DANG DOOR?!

Web Site DESIGN Co.

C'mon in!

OUR AWARDS
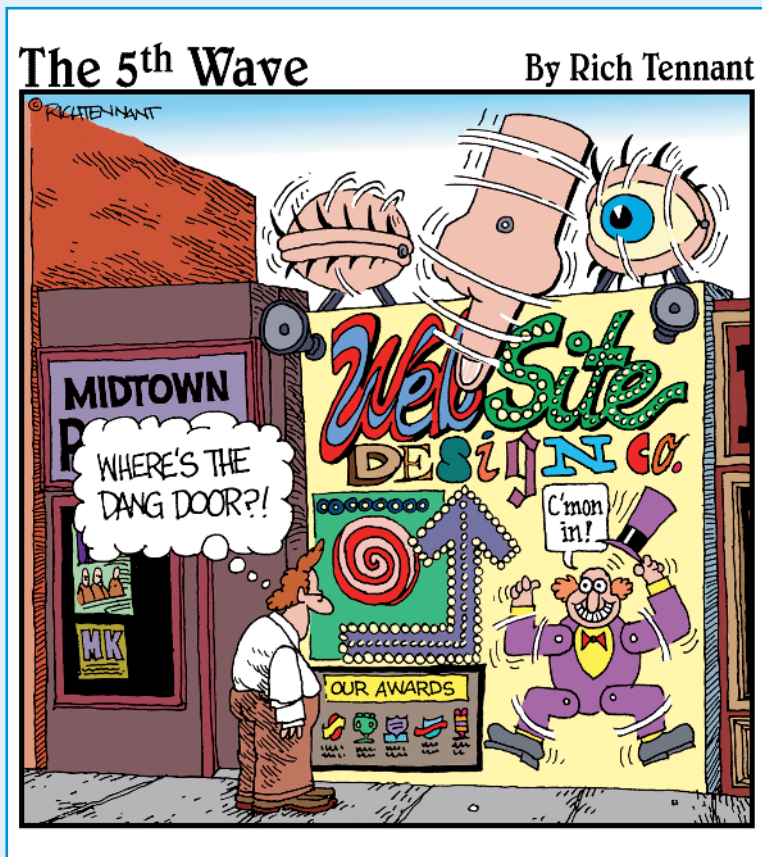
*A*t the end of the day, when all the designers go home for their well-deserved rest, the HTML and programmer folks come in — lattes in hand — and begin final assembly of the Web site. Although Web site design is a team activity, everyone on the team must know a little bit about everyone else's job in order to make the whole process go smoothly. As a Web designer, this means you must know enough about how HTML, CSS, and programming languages work so you can maximize their strengths.

For example, if you design a graphically heavy interface with a weird, slanted layout, the HTML people will have a hard time implementing the design and making it look the way you intend. So, Chapter 13 steps you through what you can do with basic HTML. Chapter 14 dives a little deeper and explores how to push HTML's design capabilities and also introduces you to CSS, which gives you a greater range of design flexibility and control. Finally, Chapter 15 shows you how to create Web applications by integrating programming languages and databases into your HTML.

By knowing how HTML and all these integrated technologies work, you will be in a better position to design Web pages that load fast, are easy to update, practical to build within a given time and budget, and look great.

13
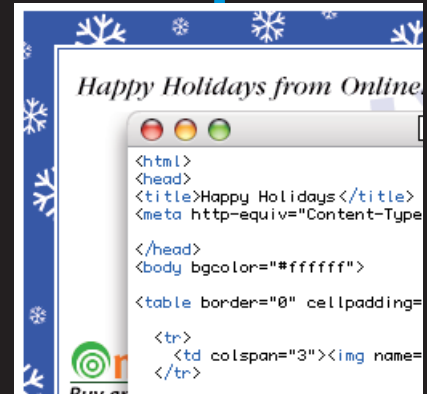
Surveying the HTML Landscape

*E*very Web page has two sides — the nice-looking side that visitors see, and the HTML side that holds everything together and makes it all work. As a Web designer, you must know how the less-glamorous flip side of your page works so you can be a more effective designer. I'm not saying that you must know how to program HTML with your eyes closed, but you do need to know how HTML is structured, how you can maximize it, and under what circumstances you can copy and paste it (my favorite form of HTML coding).

Mastering HTML is really not that difficult. The Web itself makes learning HTML tricks, tips, and shortcuts from other Web pages easy because your browser enables you to take a sneak peek at the HTML coding behind any Web page. In addition, the many powerful HTML software tools available make assembling whole Web sites as easy as using a graphics program to build your graphics — in fact, many of these software programs' interfaces look similar. In this chapter, you discover how to use tables, frames, and CSS to effectively control your layout, how to use background tiles in your Web page, and how to make your page interactive with links.

So, without further ado, put on your propeller cap and take a look under the hood and into the world of HTML coding.

HTML: The Glue that Holds a Page Together

Everything that you see on most Web pages is held in place by a page layout language called *HTML* (HyperText Markup Language). Since the time that HTML was first introduced, it has evolved from a simple way to display and link documents on multiple types of computers to its current capabilities of supporting complex page layouts, rich interactive media, and add-in languages such as JavaScript.

TECHNICAL STUFF

A browser, such as Mozilla Firefox or Internet Explorer, interprets the HTML code and draws the nice-looking side of the page that we all know and love. (It's interesting to note that the first graphical Web browser on the scene, introduced in 1993, was called Mosaic. This little program was an overnight success and, just one year later, evolved into Netscape under the code name "Mozilla." So it seems we've come full circle.)

Within the browser window, HTML is sort of like the "man behind the curtain" in *The Wizard of Oz*, pulling off an amazing display of graphics, animation, and media, all without being seen — that is, of course, until you peek.

WEB SPEAK

Upon first glance at a page's HTML code, you may think that you're looking at specs for a rocket launcher, but it's really not that complex. HTML uses a system of *tags,* bits of code that define every type of element (from links to images) on a Web page. Tags are available to place images, create tables, define the font used, and just about every other task in between. All these tags are marked by a set of opening (<) and closing (>) characters. For example, an `<a>` HTML tag used for buttons and links looks like this:

```
<a href="page2.htm">Page 2</a>
```

In this code example, the user would see only an underlined `Page 2` link on the Web page. Clicking the link would take the user to a page named `page2.htm`. The link's text does not have to mirror the name of the page you are linking to. For example, the same `<a href="page2.htm">` code can work just as well for the text `Click here to see my collection of pocket protectors`. In the following example, the whole sentence would appear as an underlined link to `page2.htm`:

```
<a href="page2.htm">Click here to see my collection of pocket
        protectors.</a>
```

For the most part, tags come in pairs. Each pair has an opening tag — in this case, `<a href="page2.htm">` — and a closing tag — `</a>`. Both the opening and closing tags are contained within the `<` and `>` characters. The text in between the tag set (`Page 2`) is the text you actually see on the Web page. For example, on a Web page, this HTML code would appear like the simple text link shown in Figure 13-1.

Page 2

Figure 13-1: All links use `<a href>` HTML tags.

Learning HTML is mainly about learning all the various tags and where they go in the code structure. If you can remember this much, you can get pretty far in your copying and pasting career. You can literally build Web pages by copying and pasting tags from other pages and then modifying them slightly for your own purposes. In fact, learning HTML by copying, pasting, and modifying is encouraged in the Web industry. You must be careful, of course, to validate your code to make sure you pasted everything correctly. Dreamweaver is pretty good at highlighting code fragments that need fixing.

Sneaking a peek at the HTML source

Web browsers enable you to view the HTML source code of a page. As shown in Figure 13-2, sneaking a peek at the HTML code of a Web page is a great way to see how it's built and to see how you can do something similar.

To see the HTML code behind a Web page, follow these simple steps:

1. **Get online and open a Web browser, such as Mozilla Firefox or Internet Explorer.**

2. **Go to any Web site by typing the URL of any Web page that you'd like to see the HTML source of.**

3. **Look at the HTML source of the page:**

    • In Firefox, choose View⇨Page Source from the menu bar.

    • In Internet Explorer, choose View⇨Source.

    As shown in Figure 13-2, you see a new window with the HTML code for the current page you are viewing. If you glance through, you see the text content of the page and all the <a> tags for each link. You can select and copy any portion of the page.

The source window will just show you the HTML but not any externally linked CSS documents. (See the "Using CSS <div> tags" section, later in this chapter, to find out more about Cascading Style Sheets.) If the page uses CSS, the <head> section of the HTML code will contain the URL path to the CSS file. You can copy the path and paste that into your browser's location field, like typing in a Web address, and see the CSS code that the page uses.

Learning (borrowing) from others

Now that you know that you can peek behind the HTML curtain, the next time you see an interesting Web page when surfing around the Internet, bookmark it for future reference. I've found that the best way to teach yourself HTML is to study the structure of pages you see every day. After looking at a number of these pages and picking them apart, you start to understand the basic structure of HTML code and the order in which tags should be laid out on a page. Understanding this before you start copying and pasting HTML code is crucial.
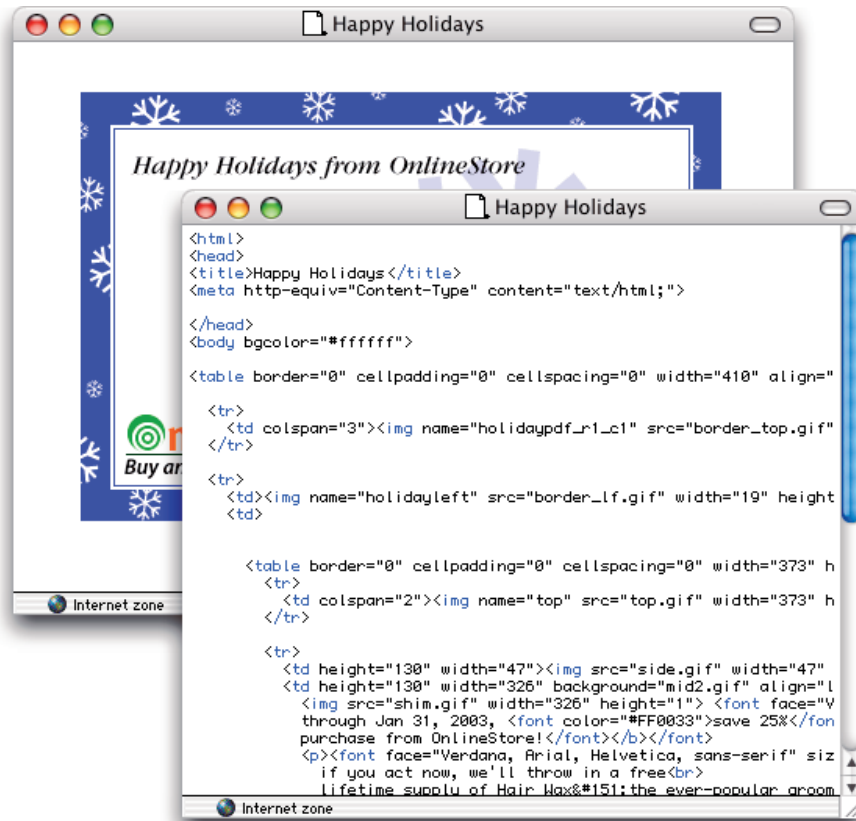
Figure 13-2: Most browsers allow you to see the HTML source code of any Web page.

TECHNICAL STUFF

Like the wings of a butterfly, opening and closing HTML tags mirror each other in their position on the page. For example, the first line of an HTML-coded page opens with the `<html>` tag. The last line is the closing `</html>` tag. This sort of mirrored hierarchy continues even down to the object level (like sentences, buttons, and so on). In Figure 13-3, notice how each closing tag follows in the reverse order of its opening tag.

TIP

After you know the basic structure of an HTML page, you understand what parts to copy from other Web pages and where to paste them into your own page for further editing. If nothing else, copying and pasting HTML can save you a lot of typing and bugs due to accidentally leaving something out.

No two browsers interpret the same HTML code in the same way. Also, some HTML tags are unique to each browser. When a browser doesn't recognize a tag, it simply ignores it. Building a Web page that looks horrible in one browser

and perfectly fine in another is entirely possible. Take a look at Figure 13-4 and Figure 13-5, for example. In Figure 13-4, this mini holiday greeting card is all screwed up when viewed in Internet Explorer. In Figure 13-5, this same card looks fine when viewed in Firefox. The reverse scenario is also true. You can build pages that look great in Internet Explorer, but look terrible in Firefox.

Two CSS "span class" tags surrounding the sentence.

```
22
23  <span class="style1">Learn where to copy and paste and you're<span class="style2"> golden</span>.</span>
24
```

Learn where to copy and paste and you're golden.

What the user sees on the page.
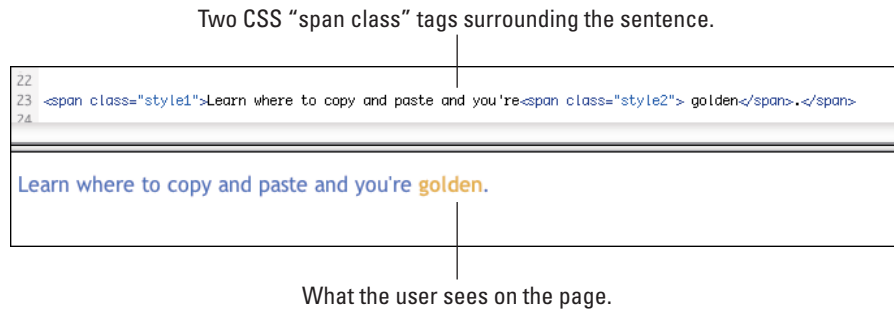
Figure 13-3: Tags mirror each other in their position on either side of the text.

Disjointed image

Misaligned text   Repeated tile

Figure 13-4: In Internet Explorer, this HTML page is stretched because the text is too large and forces the table cells to expand.

Figure 13-5: In Firefox, the same page looks perfect because the text is the intended size.

The main problem in Figure 13-4 is the size of the fonts. In this case, the fonts are much larger in Internet Explorer. The larger font size enlarges the size of the table cell, expanding the whole design. The other problem is that the cell is using a background tile. (Using a background tile is the only way to get the effect of writing HTML text on top of an image.) When the table cell expands with the larger text, you see the tile repeat at a weird place.

One way to correct the repeating tile problem is to make the tile much larger than the table cell. Table cell background tile images are the only elements that don't expand the size of a cell. So, you can put an oversized background graphic in there and then specify a smaller width and height for the cell (thereby revealing only a portion of your tile). If, however, the text and graphics you place in your cell cause it to expand, the worst that happens is that you see a little more of your background tile instead of it repeating.

If you use large background tiles, just make sure you keep their designs simple and use fewer colors to keep the file size down. Otherwise, the page will take a long time to download. To fix the too-big font problem, use smaller absolute font sizes or CSS to specify point sizes. Flip ahead to the "Using CSS <div> tags" section, later in this chapter, to find out more about CSS. In any case, it goes without saying that you must always test your HTML page on a number of browsers before you put it on the Internet for all to see.

History of HTML and the Web

Though HTML and the Web as we know them today seem like they've only been around for a few short years, their roots actually trace back to the 1950s. Take a look at these interesting sites for a catalog of the Web's evolving history:

    www.w3.org/

The World Wide Web Consortium is sort of the keeper of the World Wide Web kingdom —

approving all new HTML standards and just about every technology that integrates with HTML.

    www.isoc.org/internet/history/
       brief.shtml

The Internet Society (ISOC) site has a great timeline that traces the idea of networked information.

Using Frames, Tables, and Div Tags

By itself, HTML likes to arrange content from top to bottom and from left to right on the page. You can't get the effect of multiple columns or rows like a newspaper without using some sort of container system. Fortunately, three HTML container-like constructs are available to help you segment the page and to give you a lot more design flexibility: frames, tables, and CSS `<div>` tags (short for *division*).

Frames

Now with CSS taking over as the new standard, frames are a bit old-school. Nevertheless, they do have interesting properties that can come in handy. Frames are similar to a window that contains several panes, each pane being in the shape of a square or a rectangle. Like a window, a Web page that uses frames is the *parent* that organizes multiple *child* Web pages within each of its *panes* (or frames). Inside each frame is a whole new Web page. This versatility enables you to design independently operating segments of your page. For example, one frame may contain a long scrolling list of content, whereas another frame with navigational choices does not scroll. This way, your navigation choices don't scroll away off screen along with the content.

WARNING!

When you use frames, you chop the original Web page into multiple pages. You can use as many frames as you want to break the original Web page into multiple Web pages, but if you get too crazy, the Web page can become too difficult to manage.

When you click a link in one frame to change the stuff in another frame, the link has to *target* that frame. If the link does not target another frame, the new page loads into the current frame. Before long, you have a mess on your hands. Basically, when using frames, you must also pay special attention to the way you set up links and make sure to direct them to specific locations.

Iframes

Iframes, short for *inline frames*, are a cool variation on the frame concept. While most Web developers snub their noses at the notion of using frames for reasons I mention in a moment, iframes enjoy a little more acceptance. Unlike regular frames whereby the entire Web page is carved into a matrix of individual frames, with iframes, you can place a single embedded frame anywhere you like on the page, just like an image. My watercolor painting site at www.lopuck.com uses iframes in the Gallery section as shown in Figure 13-6.

www.lopuck.com

Figure 13-6: Selected paintings load in and out of the iframe. This way, you don't need to reload the whole page and navigation each time you select a painting. Just the iframe updates.

There are downsides to using frames that you should be aware of. For one, many professionals feel it is more difficult to maintain and update content in a frame-based site, so frames are not recommended for large scale sites. Secondly, frames can present an accessibility problem. Assistive technologies can have trouble navigating frame-based sites. Lastly, some browsers can interpret the frame layout differently, which can mess up the visual design of the site.

Tables

Another way to segment content on the page is to use tables. Unlike frames, tables don't segment the page into multiple window panes. Instead, tables go *on* a Web page like an iframe. A *table* is like a spreadsheet that enables you to put different content into each cell of the table's grid. Tables give you tremendous control over the page layout because you can slice the table grid any way you like in order to create a number of cells. Each cell is like a mini Web page.

As illustrated in Figure 13-7, each cell can have its own background color or tile, text, graphics, and form elements. In Figure 13-7, you can see the diagram of the table's structure and then how the table is filled with different form elements in each cell (text, text fields, a graphic, and a button) to create an interesting layout.

Diagram of underlying table structure

Elements placed inside the table's cells
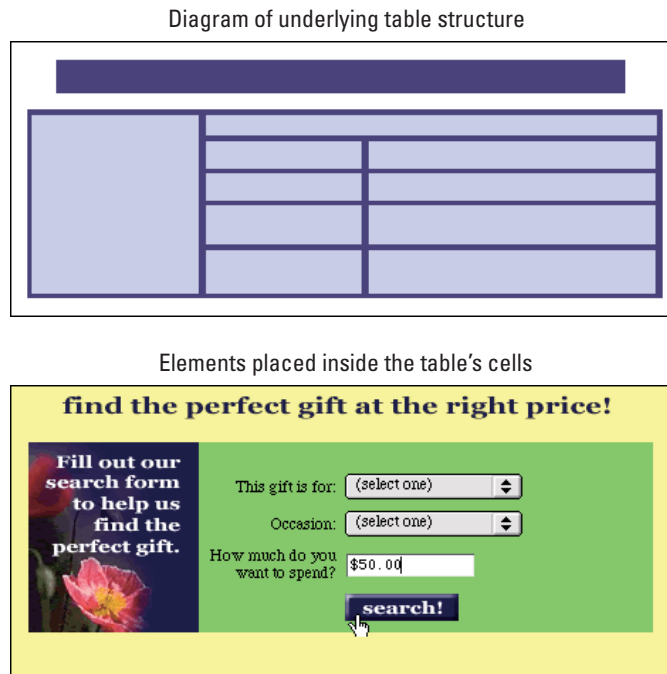
Figure 13-7: By placing different design elements in different table cells, you can control their alignment.

You can even place another whole table inside of a cell. Placing tables inside of table cells (called *nesting* tables in the biz) gives you an amazing amount of layout control. The danger of using nested tables, however, is that all content in the innermost nested tables must be fully downloaded for the outermost table to display. Additionally, nested tables present accessibility problems in terms of the order in which assistive technologies read content back to the user.

Using CSS <div> tags

The modern way of placing content on a Web page is to use CSS (Cascading Style Sheets) <div> tags. Each <div> tag is a *division* of the Web page that can contain any HTML element such as a table, form, graphics, Flash movie, or other elements. Each <div> tag is essentially its own Web page. What's more is that you can define numerous <div> tags on one Web page and stack them on top of each other, nest them inside each other, or overlap them for interesting effects. For this reason, <div> tags are usually referred to as *CSS layers*.

You can control the order, or *z index*, in which the layers stack by defining such an attribute in the <div> tag as follows:

```
<div id="NewLayer" style="position:absolute; left:128px;
        top:70px: width:400px; height:300; z-index:1">New
        Layer</div>
```

Lower z-index numbers appear underneath higher z-index numbers. Figure 13-8 shows some interesting typography effects you can achieve by overlapping different <div> layers each with different HTML text.

Figure 13-8: Overlapping CSS layers, each with a different type treatment, begin to show the design possibilities and freedoms that <div> tags allow.

TIP

Macromedia Dreamweaver has substantial support for CSS and makes building Web pages with layers and CSS styles quite easy. You can click and drag to define a new `<div>` tag, and then proceed to add HTML elements inside it. When through, a handle in the upper left allows you to drag it around on-screen for positioning.

Letting HTML Do the Design Work

By using HTML features to their fullest, you can gain a lot of creative control over the layout of your Web page. In addition to enabling you to control the layout, HTML features (such as tables, CSS styles and `<div>` tags, background tiles, and background colors) can also contribute to the visual design of your page. By using colored tables, for instance, you can add visual effects while cutting down on the need for graphics.

By using fewer graphics and more HTML, you can enhance the performance of a Web site. In addition, by relying less on graphics and more on HTML, you make the page easier to update, translate into other languages, and automate with dynamic HTML technologies.

Coloring with tables

You can create a lot of visual effects just with tables alone. For example, each table cell can have its own background color that you set with HTML tags. In addition, each table can have a border that's any color you choose.

Take a look at the banner graphic in Figure 13-9. This rich design actually uses no graphics at all. The whole thing is done with a fairly complex HTML table, as shown in Figure 13-10. By simply coloring a bunch of cells with the `bgcolor` attribute, I was able to achieve the multicolor scheme.
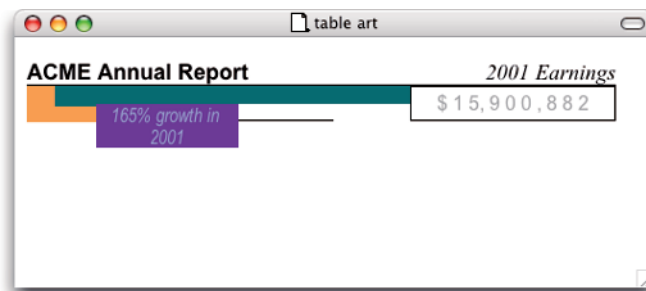
Figure 13-9: This design contains no graphics — just colored table cells and text.

Figure 13-10: The table structure for Figure 13-9.

Designing with background colors and tiles

CSS allows you to have a lot more control over the use of background images, which I explain in the next chapter, but here's what you can do with plain old HTML. With HTML, you can specify a background tile and a background color for an entire page. These provide two ways to add texture to each page before you even put anything on it. It's like buying preprinted paper for your printer.

Plus, if you use the same background tile throughout a Web site, you can take advantage of the browser's caching ability. After the background tile loads once, the browser stores it in memory. The next time it appears, the browser quickly redraws it from memory — a much faster process than downloading it again.

TECHNICAL STUFF

Adding a background tile and a background color to your HTML page is simple. You just need to add two tags in the <body> section of the page like this:

```
<html>
<head>
<title>Using Background colors and tiles</title>
</head>

<body bgcolor="#FF9900" background="tilename.gif">
</body>

</html>
```

In the <body> tag, the background color is defined with the bgcolor attribute. The number that follows, #FF9900, is the hexadecimal number for the color of the page. Following bgcolor is the background attribute that points to the filename of the art needed for the tile (in this case, tilename.gif.) (Keep in mind that for defining visual elements, HTML is going away in favor of using CSS. HTML will ultimately be relegated to just being the container.)

REMEMBER

When you use background tiles, remember to take the word *tile* seriously. The nature of tiles is that they repeat and create a pattern. You can make your tile as big or as small as you'd like. In fact, a lot of designers like to make interesting designs that are much larger than the Web page. This way, you have a lot of control over the design, and it won't repeat as a pattern until it's way off the screen where the user is not likely to see it.

In Figure 13-11, look how much larger the tile is than the viewable space of the browser window. Because this design is fairly monotone, I can compress it down to just 25K.
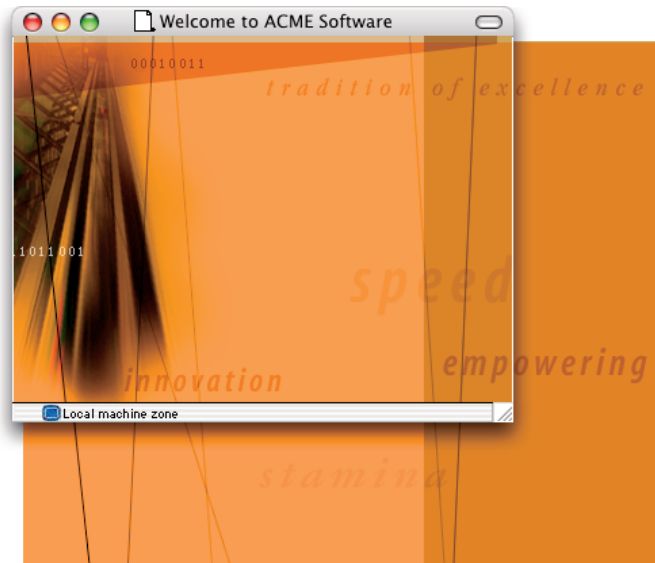
Figure 13-11: A background tile much larger than the Web page repeats its pattern where the user is not likely to see it.

Making Your Pages Interactive

HTML, on its own, is capable of a few types of links that make a page interactive, as I show in the following sections.

Regular links

You can assign a link to any selected graphic or text element. You simply wrap the element with an `<a>` tag as follows:

```
<a href="page1.htm">Text link</a>
<a href="page2.htm"><img src="image.jpg" width="100"
          height="80" border="0"></a>
```

In the first case, users see an underlined `Text link` link on the page. Clicking it takes them to `page1.htm`. In the second case, users see the cursor change to a pointing hand when it rolls over the image. If they click it, they go to `page2.htm`.

Image maps

Normally, one image can have only one link associated with it. If you want to assign multiple links to an image, you must use an image map. An image map has two components:

✔ The attribute `"usemap="#nameofmap"` that is appended to the image tag

✔ The image map tag that contains the coordinates of the various links

Here is an example of the code:

```
<img src="image.jpg" width="100" height="80" border="0"
         usemap="#nameofmap">

<map name="nameofmap">
<area shape="rect" cords="40,50,130,120" href="page3.htm">
<area shape="rect" cords="20,15,60,35" href="page4.htm">
<area shape="rect" cords="7,50,45,100" href="page5.htm">
</map>
```

The coordinates are measured in pixels from the top-left corner of the graphic. An image map needs two sets of X and Y coordinates to define the clickable region, the upper-left corner where it starts and the lower-right corner where it ends. In this code sample, the first set of coordinates say `40,50,130,120`. That means the first clickable region starts 40 pixels from the left edge and 50 pixels down from the top and ends in the lower-right corner 130 pixels away from the left edge and 120 down from the top. See the diagram in Figure 13-12.

Figure 13-12: Two points on the XY axis are needed to define a clickable region of an image map — the upper left and the lower right.

Anchor links

Rather than jumping you to a different Web page, an *anchor link* jumps you to a specific point that you define on the current page.

Anchor links are useful when you have long scrolling pages and you want the user to get to the right section quickly. For example, FAQ pages typically have a list of question links at the top of the page. Clicking a question jumps you down the page to the answer. A Back to Top link takes you to the list of questions again.

To make an anchor link work in HTML, you need two things: a link to an anchor and an anchor point. An anchor link looks pretty much the same as a normal link, but instead of typing the name of a Web page, such as **page1.htm**, you type **#** followed by the name of the anchor:

```
<a href="#part1">Part I</a>
```

The anchor point is created by using the link tag with the `name` attribute at the spot in the Web page you want to jump to:

```
<a name="part1"></a>Part I: Part one text is found here.
```

E-mail links

HTML can link to someone's e-mail address. E-mail links look just like any other links on the page, but usually when you click them, the browser opens an e-mail program with a message preaddressed to the intended recipient.

E-mail links use the `mailto` attribute inside the link tag like this:

```
<a href="mailto:lopuck@lopuck.com">Email me</a>
```

Aside from regular links, image maps, anchor links, and e-mail links, HTML can't do much more by itself to make a page interactive. You can define form elements, such as buttons and text fields, but to make them do anything worthwhile, such as process a credit card order, you have to sprinkle in some programming written in other languages like ASP (Active Server Pages), ColdFusion, or JavaScript. These kinds of languages weave right into the HTML code to supercharge the interactive experience. For more discussion on programming languages integrating with HTML, see Chapter 15.

Coding to Make You Feel Proud

Now that your head is spinning with all the things you can do with HTML coding, the good news is that you don't have to write a smidge of HTML by yourself if you don't want to. A number of *WYSIWYG* (What You See Is What You Get) software tools on the market allow you to code like a true techie,

and no one ever has to know that you're just a designer. (A WYSIWYG editor is a software program like Dreamweaver, my personal favorite, that enables you to build Web pages visually by clicking icons and making menu selections instead of typing code into a text editor.)

What's more, a lot of these HTML-writing tools have interfaces similar to those in graphics programs that you may already be familiar with, such as Fireworks and Photoshop. If a WYSIWYG editor is not your style, a few HTML text-only editors out there can put a few hairs on your chest — even if you're a woman!

You should know that there are two schools of thought on using WYSIWYG HTML editors. Some hard-liners think WYSIWYG editors unnecessarily muddy up the code. Other people think WYSIWYG editors are a great tool that can help you speed up the Web page production process and create Web pages without forcing you to know all the details of HTML coding. I'm one of the latter folks.

HTML editing with power tools

If you're like me, you want as much help as you can get when you're building a Web page. I have no pride when it comes to using a WYSIWYG HTML editor if it saves me from typing HTML code. I am more than happy to click a little icon that inserts a whole table structure — and gladly take credit for that little maneuver.

Dreamweaver is the most evolved and established of the Web site building tools. Dreamweaver also is customizable in the sense that you can buy functionality plug-ins like eCart from WebAssist that allow you to step through the process of building an online shopping cart.

WebAssist also has a Web design add-on to Dreamweaver called SiteAssist, shown in Figure 13-13. Once installed into Dreamweaver, SiteAssist allows you to create a whole Web site literally in seconds by first choosing a site type like Club/Association or Corporate. Each site type has a site tree that includes all the typical pages you'd want in such a site. You then choose from its design and navigation styles, and mix and match these with a number of color schemes for truly endless design combinations. Your only job after selecting these preferences is to populate the new site with content.

Using a text editor: Commando-style HTML

For you code warriors out there who want to type each line of HTML code yourself, you can use almost any text editor. Some text tools, however,

are specifically built for writing code. Some of these tools, such as BBEdit (a text editor that you can purchase online) for the Mac or HomeSite for the PC, number each line for easy reference and offer some nice features, such as Find and Replace. You can also hand-code HTML in Dreamweaver with the added benefit of using a split-screen view mode where you can see how your code looks in one window while you type it in the other.

In addition, hand coding HTML in a text editor or right in Dreamweaver enables you serious programmers to build forms and other common HTML elements quickly for squeaky clean, custom code.
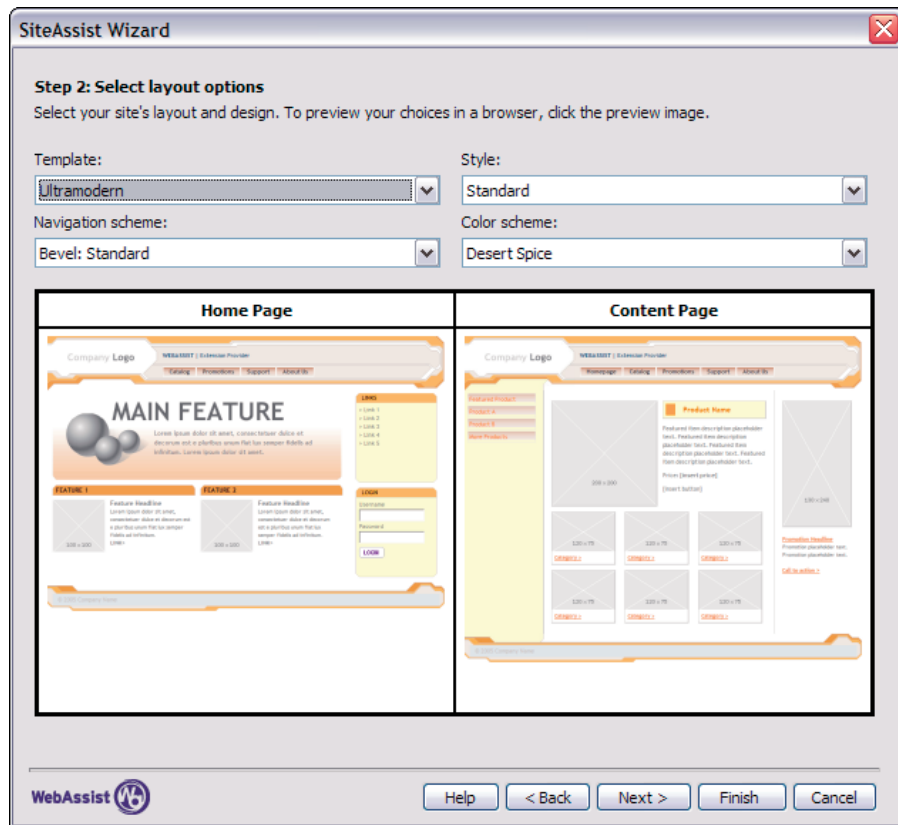
www.webassist.com

Figure 13-13: WebAssist's SiteAssist product allows you to build an entire Web site in seconds, including the design, navigation, and site structure.

14

Controlling Page Layout

*H*TML was originally designed by gear-heads who were thinking less about design and more about the practical aspect of linking pages of text together. As a result, a lot of page design features were afterthoughts. Because designers have become a part of the Web industry, HTML has evolved to include a few more page layout control features, such as the ability to adjust table elements and the robust control afforded by Cascading Style Sheets, or *CSS*, which allow you to place stuff wherever you want and apply formatting to text and graphics.

In this chapter, you find out how to work with tables more effectively and understand how a few common CSS techniques can help you gain more control over your Web page designs. After you finish laying out your pages, you can save them as HTML page templates that the production team can use throughout the site for a consistent presentation of the content.

Fixed-Width and Stretchy Tables

You can approach the width of a table in two ways. The table can always be a fixed width, say 800 pixels wide to accommodate the current standard for designing Web sites, or it can expand and contract with the width of the user's browser window.

While creating a stretchy table that expands and shrinks is a great way to accommodate all browser sizes, it adjusts the layout of your pages. For example, if you have a text column, at times it's super narrow and at other times too wide for a designer's taste. See the possible scenarios illustrated in Figures 14-1 and 14-2.
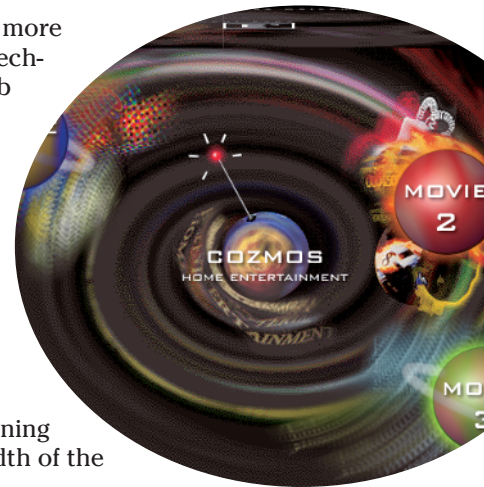
Figure 14-1: In a "stretchy" table, the content stretches to fit the width of the browser window.

The unpredictability of the browser window size is always something you should keep in mind as a designer.

For example, if a user has an extremely large screen like the Apple cinema display, a fixed-width design can look funny. Your Web site looks like a postage stamp up in the corner surrounded by a bunch of white space (unless you have a background image strategy in place). Alternatively, a stretchy design could look so stretched out that the net effect is worse.

Here are some design thoughts to mitigate the unpredictable nature of browser window size:

- **Try centering your Web site design.** If you are using a fixed-width design, at least it will float in the center of the browser window instead of looking like a postage stamp clinging to the upper-left corner of the window.

- **Think through a background image strategy for your site.** Create an interesting background tile or images and use CSS to place or tile it as needed to create the right effect. That way, as a designer, even if your site is fixed width, you are accommodating larger browser windows and creating a seamless, well-thought-out design.

- **Set your table to a percentage of the browser width.** If you're using a stretchy table, you can set the width to be a percentage of the browser window's width. For example, if you set the table width to 80%, your content still shifts around inside the table, but at least there is always a right margin or, if centered, a margin on either side.

- **Design your layouts with the stretch in mind.** As shown in Figures 14-1 and 14-2, the layout was designed in such a way that the content looks good in any size browser window.

Figure 14-2: If the user squishes the browser window, the content inside shifts around to fill the new window size.

Precise Positioning with CSS Layers

Although tables afford you a lot of layout control, placing things exactly where you want them to go within the table (and thus on the page) can be challenging. To solve this problem, you can use CSS <div> tags, also called *CSS layers*, either on their own or in combination with tables (such as placing a table inside a <div> tag). When you use a CSS layer, you basically create a mini–Web page within which you can place text elements, buttons, graphics, and any other HTML element such as tables.

TIP

CSS uses a whole system of tags and code syntax. For this reason, my advice is to use a Web authoring tool such as Dreamweaver to write the basic CSS code for you and then become familiar with all the various CSS properties so you can make these updates by hand. My two favorite online CSS resources are W3 Schools (www.w3schools.com) and A List Apart (www.alistapart.com).

To position a CSS layer on a Web page, enter specific X and Y coordinates. For example, if X equals 25 and Y equals 25, the upper-left corner of the CSS layer starts 25 pixels down from the top and 25 pixels from the left edge of the page.

If you're using Web authoring software, you can simply drag a visual representation of the CSS layer anywhere on the page, and the software automatically keeps track of its X and Y position and updates the CSS code in the HTML. If you're creating a CSS layer from scratch, the X and Y information is an attribute listed inside the CSS <div> tag as follows:

```
<div id="Layer1" style="position:absolute; left:70px;
        top:50px; width:150px; height:130px; z-
        index:1">Insert graphics, text, or whatever
        here</div>
```

TECHNICAL STUFF

In this example, the CSS layer named Layer1 is positioned 70 pixels from the left edge of the page and 50 pixels down from the top edge. I also want to call your attention to the z-index attribute. This attribute controls how the layer falls on top of or below other CSS layers. Layers with higher z-index numbers overlap those with lower z-index numbers. This capability enables you to achieve the unique effect of overlapping text and graphic elements, as shown in Figure 14-3.

While this example uses a fixed absolute position, you can also specify a relative position. The following code offsets the <div> tag 20% from the left side and 10% from the top side of the browser window. As you widen or shrink the browser window, the <div> tag always starts at the 20% mark.

```
<div id="Layer1" style="position:relative; left:20%; top:10%;
        width:100px; height:200px; z-index:2">Insert
        graphics, text, or whatever here</div>
```

Figure 14-3: CSS layers enable you to place elements anywhere you want on the page — even on top of other elements.

WARNING!

CSS layers are not without their problems. Older browsers — Netscape Navigator in particular — can have trouble interpreting them. If the user resizes the browser window after a page with CSS layers loads, the layers may shift all over the page, and some of your text formatting may be lost. The user must reload the Web page (using the browser's Reload or Refresh button) to correct the layout. Interestingly, Dreamweaver has a special fix for this under the Commands menu called Add/Remove Netscape Resize Fix.

Also, something to keep in mind is that different browsers may shift the placement of positioning, padding, margins, and borders slightly making your design not always as precise as you wanted.

Controlling Design Elements with CSS

Aside from creating layers that you can position anywhere on the page, CSS has an extensive array of design options you can apply to text and graphical elements. For example, you can create a CSS style to specify a particular font, font size, color, and paragraph alignment for any HTML text element, such as a header, a caption, and so on.

There are three kinds of CSS style sheets:

✔ **Internal:** An internal style sheet is when all the CSS styles are defined in the `<head>` section of a Web page and are used only on that page.

✔ **External:** An external style sheet is a separate document — apart from your Web page — that contains all the CSS styles. That way, multiple Web pages can all reference a single CSS document for all style information, and if you ever want to make a global change, you can change the single CSS document, and all Web pages that reference it update automatically.

✔ **Inline:** A CSS style can be defined inside a single HTML element on the page. For example:

```
<p style="color: a60433; font-weight: bold; font-size:
        xx-large; text-align: left;">This is an inline
        style applied</p>
```

The "cascade" aspect of Cascading Style Sheets refers to its ability to draw style information from all three types of CSS definitions and blend them together. The order, or "cascade," of each style definition determines which one takes precedence when an object is subjected to multiple styles. The order of priority is first an inline style followed by an internal style sheet, followed by an external style sheet, and lastly, if all else fails, by the browser default settings.

Page margin control

By default, an HTML page leaves about a ten-pixel margin from the top and left border of the page for content placed on a Web page (excluding background tiles and background page colors). You can, however, set the margin to whatever you'd like. You can make your page start content in the upper-leftmost corner at 0, 0 (X,Y coordinates), or you can leave a large margin like the one shown in Figure 14-4. In this example, I used a decorative background image and set the page margin so that all the content falls on the large colored block of the image.

The preferred method of setting page margins is to use CSS. In the `<head>` section of your HTML code (in between the opening and closing `<head>` tags), you can add the following CSS code:

```
<style type="text/css">
body {
        margin-left: 50px;
        margin-right: 50px;
        margin-bottom: 50px;
}
</style>
```

Figure 14-4: The background tile and the page margin work together to create a layout.

Smarter still is to define the page margins in an external style sheet. That way the margin setting would apply to the whole site. If one page needs to be an exception, you can use an internal CSS margin setting, which, due to the cascading nature of style sheets, will take precedence over the external CSS margin setting.

TIP

For maximum browser compatibility, be sure that you do not leave any spaces between 50 (or whatever your number is) and the px (pixel) value. Firefox and Netscape have issues with a space before the px.

Background images and tiles

In Chapter 13, I discuss the normal HTML properties of background tiles. They start at the upper-left corner of the page and repeat end to end and top to bottom. To avoid the repeating nature of background images, and to gain a lot more design control over background images in general, you can use CSS.

Repeating patterns — or not

The following is the snippet of CSS code you'd insert in the <head> section of your HTML page to center an image in the background of your page, and not have it repeat as a pattern:

```
<style type="text/css">
body
{
background-image: url('tile.gif')
background-repeat: no-repeat;
background-position: center;
}
</style>
```

The `background-repeat` property in this example is set to not repeat in a tiled pattern. If you do want it to repeat, you simply enter `repeat` for the value. You can also control whether the image repeats only down the X or Y axis of your page. To do so, you use this syntax:

```
background-repeat: repeat-y;
```

Fixing the background so it doesn't scroll with the page

If you want the background image to remain in a fixed position on the page while the rest of the content on the page scrolls, add the following attribute to the list of attributes (add it just after the `background-repeat` property):

```
background-attachment: fixed;
```

Background tile positioning

Finally, you can decide where the background image should start on the page. The first line of code is the complete example. Following the example are a list of other values that you can substitute for `top left`.

```
Background-position: top left;

Top center;
Top right;
Center left;
Center center;
Center right;
Bottom left;
Bottom center;
Bottom right;
40% 60%
50px 50px
```

The last two lines are expressed in real numbers. I chose 40 and 60%, and 50 pixels as examples only. As you can see, CSS offers tremendous control over placing design elements where you need them on your page.

Setting background colors

With CSS, not only can you set the background color for a Web page (like you can do with standard HTML tags), you can also set the background color for text elements. The effect of setting the background color of text is sort of like a highlighter marker. The code for the example shown in Figure 14-5 is as follows:

In the <head> section of your page, define a background color name like highlighter or whatever you want to call it:

```
<head>
<style type="text/css">
span.highlighter
{
background-color:yellow
}
</style>
</head>
```

Then, in the <body> section of your page, where the text lives, you can append the highlighter CSS formatting to a portion of the text as follows:

```
Don't forget to<span class="highlighter"> bring your own
          lunch</span> on Tuesday when you come.
```



```
 6  <title>Setting Text Background Colors</title>
 7
 8  <style type="text/css">
 9  span.highlighter
10  { background-color:yellow }
11  </style>
12  </head>
13
14  <body>
15  Don't forget to <span class="highlighter">bring your own lunch</span> on Tuesday when you come.
16  </body>
17  </html>
18
```

Don't forget to bring your own lunch on Tuesday when you come.

Figure 14-5: You can define the background color of text elements with CSS.

Text handling

CSS gives you tremendous control over text. You can make text any color, create borders around all sides of it, set its background color so it looks like you used a highlighter; the list goes on and on. CSS is so deep that plenty of books are available on the subject. Just to give you the sampler plate, here are a few frequently used options along with some code samples:

Setting text font, color, size, and style

Like you can with HTML, CSS allows you to set basic attributes for your text. In the `<head>` section, you first define the style.

Figure 14-6 illustrates the following code:

```
<style type="text/css">
.styleA {
        font-family: Verdana, Arial, Helvetica, sans-serif;
        font-size: 12pt;
        font-style: italic;
        line-height: 20pt;
        color: #3366CC;
}
</style>
```

In the `<body>` section, you apply the style to a segment of text:

```
<p class="styleA">This paragraph uses "style A," a collection
        of font settings. Notice the font is set to
        Verdana, the size is 12 points, it's blue, italic,
        and has a "line-height" (space between each line
        of text) set at 20 points.</p>
```

This text uses "style A," a collection of font settings. Notice the font is set to Verdana, the size is 12 points, it's blue, italic, and has a "line-height" (space between each line of text) set at 20 points.

Figure 14-6: You can set up a number of CSS styles, like `styleA` of this example, that each govern a collection of font settings.

Upper- and lowercase

As illustrated in Figure 14-7, regardless of how you capitalized the text in the HTML code, you can force text into all uppercase, all lowercase, or have initial caps. In the `<head>` section, add the following code:

```
<style type="text/css">
p.allcaps {text-transform:uppercase}
p.lowercase {text-transform:lowercase}
p.initialcaps {text-transform:capitalize}
p.smallcaps {font-variant: small-caps}
</style>
```

In the `<body>` section, add these tags around the text you want to affect:

```
<p class="allcaps">This text appears in all capital
        letters.</p>
<p class="lowercase">Even though I have Capitals, all appears
        in Lower Case.</p>
<p class="initialcaps">only the first letter of each word is
        capitalized.</p>
<p class="smallcaps">All letters are shown in small caps.</p>
```

THIS TEXT APPEARS IN ALL CAPITAL LETTERS.

even though i have capitals, all appears in lower case.

Only The First Letter Of Each Word Is Capitalized.

ALL LETTERS ARE SHOWN IN SMALL CAPS.

Figure 14-7: Even if the text in the code has mixed capitalization, CSS can force it to appear on-screen with all caps, all lowercase, or have initial caps.

Letter and line spacing

You can use CSS to loosen or tighten the *kerning* (the spacing between letters) and to control the amount of *leading* in your paragraphs (the space between each line of text). See Figure 14-8 for an example of what this `<head>` and `<body>` code looks like. Here's the `<head>` section code:

```
<style type="text/css">
p.tight {letter-spacing: -3px}
p.open {letter-spacing: 4px; font-variant: small-caps}
</style>
```

In the `<body>` section, add these tags:

```
<p class="tight">This text is too squished to read.</p>
<p class="open">Letters widely kerned and in small caps.</p>
```
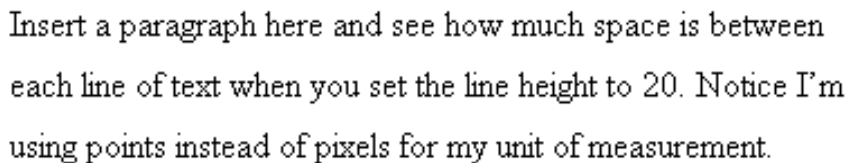

Figure 14-8: In this example, the top sentence is so tightly kerned you can't read it. In the second sentence, I combined wide kerning with small caps.

To adjust the leading of a paragraph, as shown in Figure 14-9, the syntax in the `<head>` section is:

```
<style type="text/css">
p.openleading {line-height: 20pt}
</style>
```

The syntax in the `<body>` section of the page is:

```
<p class="openleading">Insert a paragraph here and see how
          much space is between each line of text when you
          set the line height to 20. Notice I'm using points
          instead of pixels for my unit of measurement.</p>
```


Figure 14-9: Use the `line-height` property to adjust the leading of a paragraph.

Adding Extra Space Around Elements

Whether you are using frames, tables, or CSS layers to position elements on the page, oftentimes you need a little extra space here and there in between graphics, or you may just want to nudge graphics in one direction or another. To get this micro level of page layout control, you can use a number of tricks that add breathing space around your elements.

Adding margins to table cells
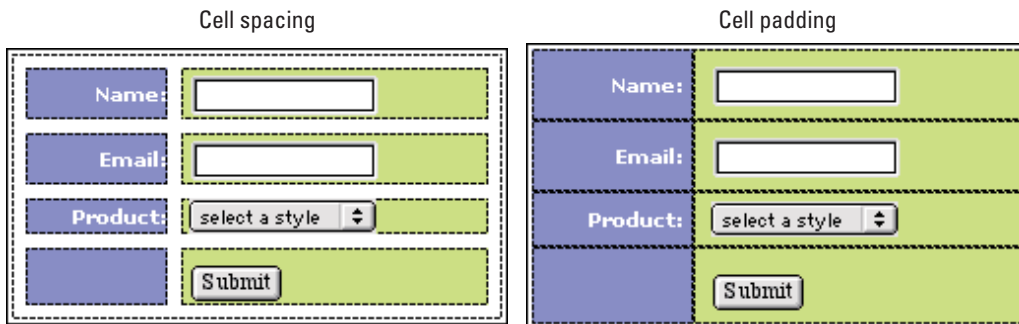
When you create a table, you have two options for adding a margin to the cells within: the `cellpadding` and `cellspacing` attributes. These two attributes are included in the opening `<table>` tag to define the table's margins. Each works a little differently, but the net effect is that each cell has a little space around it.

`cellspacing` creates a gutter between the cells. `cellpadding` adds a margin *within* each cell, so any content you add to the cell is inset from the cell's edge. See the difference between the two in Figure 14-10. As the figure shows, the two methods have a dramatic impact when you assign a background color to the cells. `cellpadding` creates no gaps between the cells themselves, however, so their background colors look like one solid color. If you want to get the effect of a continuous background color between cells and want the content within the cells to have a margin, use `cellpadding` but no `cellspacing`.

Figure 14-10: Cell spacing leaves a gap between cells, but the content is flush to the cell's edge. Cell padding adds a margin within each cell.

WARNING! You should always include both attributes in your opening `<table>` tag, even if you enter a value of zero for both. If you leave one or both of them out, the table uses its default settings: cell spacing of two pixels and cell padding of one pixel.

TIP Incidentally, if you use a table structure to piece a graphical image back, make sure that you set both cell spacing and padding to zero. As shown in Figure 14-11, this makes the image look like one seamless image. (You would cobble an image together rather than insert the image in its entirety so you can cut up the image and save each piece individually as a GIF or a JPEG. This practice, called *slicing,* allows you to get the best quality for each section of your image. For more information on slicing graphics, see Chapter 12.)

Adding space around graphics

In the course of adding graphics to your HTML page or even within a table cell, you occasionally need to insert a little space here and there so that your images and text elements are not too close together. You may also want to nudge graphics a little to the left or to the right. In this section, I share various techniques that you can use to add that little bit of extra space where you need it.

This image is broken into four table cells.

These two are GIFs.

These two are JPEGs.

The design looks seamless in a browser.

Figure 14-11: This table uses no spacing or padding, so each piece butts up perfectly to one another.

Soft and hard line breaks

When you type a line of text in HTML, the line, when it displays in a Web browser, goes all the way across to the end of the page or table cell and then

automatically wraps to the next line. Left to its own devices, the text may look funny because, as in the case of the long headline in Figure 14-12, you could have one word left dangling on the next line.

> ## Hundreds and thousands of new products to choose from.

Figure 14-12: With no line breaks, the headline may wrap, leaving just one word on the next line.

TIP

If you want to control where the text line breaks and wraps to the next line, as in Figure 14-13, you can insert a *soft line break* with a simple <br> tag. A soft line break creates less space than a hard line break, which is used to designate the space between paragraphs. Use the <br> tag like this:

```
Hundreds and thousands <br> of new products to choose from.
```

> ## Hundreds and thousands
> ## of new products to choose from.

Figure 14-13: If you insert a soft line break, the two lines now look balanced.

WARNING!

If you insert a soft line break, the page can look funny if the user shrinks the browser window. The line always breaks where you insert the <br> tag. Look what happens to the headline in Figure 14-14 when the user resizes the window. When you insert soft line breaks, you cause the page to become less fluid in its design.

> ## Don't miss our new blow out
> ## prices.
> ## Everything must go!

Figure 14-14: Using soft line breaks can cause the page to look funny when the user shrinks the browser window.

Entering a *hard line break* is like using the Enter key on the keyboard: Doing so leaves a big gap so you can start a new paragraph of text, as shown in Figure 14-15. Insert a hard break by entering the <p> tag:

```
Hundreds and thousands <br> of new products to choose from.
          <p>
Act now and you can save on all products in our new fall
          lineup. </p>
```


**Hundreds and thousands
of new products to choose from.**

Act now and you can save on all
products in our new fall line up.

Figure 14-15: By using line breaks, you can control the gap between lines and line length.

Horizontal and vertical space around graphics

You can also assign a margin around each graphical element by using the vspace (vertical space) and hspace (horizontal space) attributes in the image tag (<img>). These attributes add a number of pixels on either side of the graphic.

For example, if you assign a vspace of five pixels, you create a margin of five pixels above and below the graphic, as shown in Figure 14-16. The same principle applies for hspace. The code looks like this:

```
<img src="computer.jpg" width="168" height="140" hspace="0"
          vspace="5">
```

Using CSS to add a margin to elements

The newer standard of adding a margin to design elements is to use CSS. Rather than using the HTML vspace and hspace attributes, you can use the CSS attributes of margin-top, margin-right, margin-bottom, and margin-left. The margin attribute creates a specified pixel width gap on any or all sides of an HTML element. The following is an inline CSS example (shown in Figure 14-17):

```
<p style="color: red; margin-left: 20px">
This red text would have a 20 pixel left margin.
</p>
```

**Hundreds and thousands
of new products to choose from.**

Act now and you can save on all
products in our new fall line up.

17" Monitors | Add to order |

just **$500** while supplies last

Figure 14-16: The vspace and hspace attributes add a margin around a graphic.

Unformatted text

This red text would have a 20 pixel left margin.

Unformatted text

Figure 14-17: Use CSS to add a margin around any design element.

Building HTML Design Templates

After you massage your HTML page layouts to get them looking exactly as you want, you can save them as templates that the production team can use throughout the site. Many Web sites have hundreds of pages, but most of these pages are based on just a handful of templates. I like to distill my template pages down to the bare essentials with dummy stand-in images called *FPOs* (for position only) and greek text that are easily replaced with the real content.

In addition, if you design each template as a different layout of reusable content modules, you can gain even more production efficiencies. For example, you can design the global navigation system as one content module that is used on each template. Common elements like a system requirements module can be included on one or more template layouts. This modular approach makes updating your pages in the future easy. If you update just the navigation module, it is automatically updated throughout the site on each template that uses it.

Web Sites on Steroids

*F*inding a major commercial Web site these days that's not using some form of technological steroids to boost performance, enhance interactivity, and allow its Webmasters to automate it is rare. HyperText Markup Language (HTML) by itself can do some basic linking and control the layout presentation of the page, but that's about all. In order to do the cool stuff, such as inserting a personalized greeting or allowing people to buy products and services, you have to interweave languages such as JavaScript, VBScript, and others into the HTML code.

I put this chapter last in this part because you don't have to become a serious programmer in order to be a good Web designer (audible sigh of relief); you just need to know what kind of technological options are available to you. You must know what is and what isn't inside the realm of possibility in order to be an effective Web designer.

In this chapter, I take a survey-like approach to all the different technological options that you have available to you, what they do, and what sort of ramifications they have on your site.

Injecting Power into HTML Pages

HMTL allows you to interweave other programming languages right in to it. Without knowing any better, you could look at an HTML page, like the one shown in Figure 15-1, and not know where the HTML code stops and the programming begins.

In the example shown in Figure 15-1, Active Server Pages (ASP) programming opens a connection to an online database. The ASP code goes at the top of the page before the opening `<html>` tag. Then, later in the page, the form elements reference the ASP programming to insert the correct data in the page. The coding language and the HTML work perfectly together, leveraging each other's strengths. The ASP code grabs the data, and the HTML formats it.

REMEMBER

The one drawback to some programming languages is that if you're the tinkering designer type, you can't teach yourself by looking at the examples on other Web pages. Choosing View➪Source from the browser menu shows only the HTML code, not the programming code within. Only some languages included in the HTML page, such as JavaScript (HTML with JavaScript code in it is called DHTML, or Dynamic HTML), show up in the Source window.

The ASP code opens a
connection to a database.

```
<%
'Set ProductID
Dim PRODUCTID
PRODUCTID=38

'Set the prices
'------------------------
Dim normal, special
normal = 45
special = 40
'------------------------

%>
<html>
```

In the Web page, the embedded ASP code grabs
the correct data, and the HTML formats it.

```
<td align="right" valign="top" bgcolor="#336600" width="50">
<form method="post" action="addtoorder.asp" name="theform">

<input type="hidden" name="reg" value="0">
<input type="hidden" name="add" value="yes">
<input type="hidden" name="product" value="<%=PRODUCTID%>">
<input type="hidden" name="cost" value="0">
<input type="hidden" name="normal" value="<%=normal%>">
<input type="hidden" name="special" value="<%=special%>">
<input type="radio" name="producttype" value="normal" checked>

</form>
</td>
```

Figure 15-1: HTML and programming code can interweave side by side in the document.

JavaScript

JavaScript, also known as ECMAScript to help distinguish it from Java (an unrelated programming language), is a fairly robust, object-oriented programming language you use to add all manner of interactive features to your Web page, such as the ever-popular rollover buttons and jump menus. Because you can see JavaScript code in the browser's Source window, you can easily see how the language is used and utilize the example for your own purposes.

To use JavaScript in your Web page, define a few functions and commands at the beginning of your HTML code (before the closing `</head>` tag). Then reference these commands in the media that you place on the page. For example, if you want to make a rollover button, you first define the rollover function in the `<head>` section. In the following example, the JavaScript code creates three functions:

✔ `MM_findObj`. This function is used by the `swapImage` function to help it identify the correct image to use for the rollover.

✔ `MM_swapImage`. This function exchanges the current graphic for another that reflects the rollover state of the button when users move their mouse over the button.

✔ `MM_swapImgRestore`. The purpose of this function is to restore the original graphic, representing the resting state of the button, when users move their mouse off the button.

This code was generated automatically in Fireworks and includes `MM_` (short for Macromedia) before each of the function names. You can name your functions anything you like, however, just as long as you don't use any spaces.

After you build a rollover button in Fireworks, the program exports not only the graphics, but also an HTML page with the JavaScript code needed to make the button work:

```
<script language="JavaScript">

function MM_findObj(n, d) { //v4.01
  var p,i,x;  if(!d) d=document;
           if((p=n.indexOf("?"))>0&&parent.frames.length)
           {d=parent.frames[n.substring(p+1)].document;
           n=n.substring(0,p);}
  if(!(x=d[n])&&d.all) x=d.all[n]; for
           (i=0;!x&&i<d.forms.length;i++) x=d.forms[i][n];
  for(i=0;!x&&d.layers&&i<d.layers.length;i++)
           x=MM_findObj(n,d.layers[i].document); return x;
}

function MM_swapImage() { //v3.0
  var i,j=0,x,a=MM_swapImage.arguments; document.MM_sr=new
           Array; for(i=0;i<(a.length-2);i+=3)
  if ((x=MM_findObj(a[i]))!=null){document.MM_sr[j++]=x;
           if(!x.oSrc) x.oSrc=x.src; x.src=a[i+2];}
}
function MM_swapImgRestore() { //v3.0
  var i,x,a=document.MM_sr;
           for(i=0;a&&i<a.length&&(x=a[i])&&x.oSrc;i++)
           x.src=x.oSrc;
}
</script>
```

After that mess is out of the way, you can then reference each function by name (for example, `mm_swapImgRestore`) in the HTML code. In the following example, `button.gif` is replaced with `button2.gif` (the highlighted state) when the mouse pointer rolls over it. The `SwapImage` function takes care of the

rollover by making `button.gif` disappear and replacing it with `button2.gif`. When the user's mouse pointer rolls off the graphic, the `SwapImageRestore` restores it to its original state by displaying `button.gif` again. Notice that the reference to the JavaScript code goes inside the `<href>` link for the `button.gif` graphic and passes parameters to the function:

```
<a href="link.htm" onMouseOut="MM_swapImgRestore();"
            onMouseOver="MM_swapImage('button','','button2.gif
            ',1);" ><img name="button" src="button.gif"
width="60" height="60" border="0"></a>
```

JavaScript comes in two forms: client-side and server-side. In *client-side* JavaScript, like the previous example, the Web page itself contains all the JavaScript functions needed to work properly. Pages with *server-side* JavaScript must reference code on a remote server in order to function properly. The Web page by itself does not work correctly.

Although server-side JavaScript sounds like a pain, you can actually do cooler things with it, such as access a database to personalize the page or automatically insert information. Server-side JavaScript is ideal for creating applications that are cross-platform (UNIX, Mac, and Windows) and cross-browser compatible.

Embedded media

If you can't build your envisioned combination of interaction and design with HTML or programming languages, you can probably create it with Flash. In addition to Flash's well-known animation capabilities, it also has its own JavaScript-like scripting language called ActionScript that allows programmers to accomplish some amazing combinations of animation and interactivity. Imagine, for example, a visual drag-and-drop shopping cart system where you drag the things you want to your cart and see the price adjust accordingly, or visually build outfits with accessories, change colors and sizes, and add the whole ensemble to your cart.

ActionScript can also connect to databases and content management systems to build truly dynamic applications. These fully featured Flash applications are then dropped into your HTML page with a simple `<embed>` tag like this:

```
<embed src="movie.swf" quality=high
            pluginspage="http://www.macromedia.com/shockwave/
            download/index.cgi?P1_Prod_Version=ShockwaveFlash"
            type="application/x-shockwave-flash" width="500"
            height="350" loop="false">
</embed>
```

After you place a media element in the Web page, it has its own set of controls and acts independently of the other elements on the page (text, links, buttons, and so on). Take a look at the Flash application in Figure 15-2. In this presentation, users can design their own Mini Cooper car from the ground up, price it out, and find a local dealer.

WARNING!

Users must have a special plug-in downloaded and installed in their browser in order to view embedded media properly. For example, users who don't have the QuickTime Player plug-in get an error when loading the page. You can plan for this possibility by adding a JavaScript function to the page that auto-detects the plug-in. If the plug-in is not found, the JavaScript code could automatically redirect users to another version of the Web page that does not require a plug-in, or direct users to a URL where they can download and install the correct plug-in.

www.miniusa.com

Figure 15-2: This all-Flash site allows you to design your own Mini Cooper.

Streaming media

Although video and audio can add a lot of life to your Web site, they can be cumbersome for users to download. While many people have fast home DSL connections, many users today still access the Internet via their phone line and their computer's internal modem. The speed of these internal modems is usually 56,000 bits (56K) per second — meaning that they can download only 56K worth of data each second. If you consider that a Web page with media like a QuickTime movie can add up to one megabyte (which translates to 8,000,000 bits), the page would take nearly 2½ minutes to download. That's almost the same time it takes to sit through a TV commercial break.

To get around the problem of inadequate bandwidth, video and audio for the Web are prepared such that they *stream* in instead of download. Streaming media downloads in chunks so that people can start watching or listening almost immediately to the first chunk while the rest of the chunks download.

Working with and preparing streaming media is similar to preparing other media, such as graphics, for the Web. You must compress and save it in a Web-friendly format. This process is called *encoding* and involves selecting a combination of file format and *codec* (short for compressor/de-compressor). Sometimes the two are inextricably linked as in the case of Real Video and Windows Media — both of which are formats with their own proprietary codecs.

For optimal Web delivery, you must encode video and audio media with a target bandwidth in mind (like DSL or 56K). The lower the connection speed, the more compression you need. For even better performance, you can upload the finished files to different servers. If a lot of people look at the same file simultaneously, you don't put undue strain on one server or slow down the media's delivery if the file is distributed.

Creating Dynamic, Database-Driven Web Pages

To be a true business machine, a Web site has to act as the bridge between the customer and the company's processes. The Web site must be able to funnel a customer's requests to the right company databases so that the company can serve them. For example, customer service inquiries must route to the customer service database, whereas ordering info must first be processed and then sent on to both the shipping and accounting departments.

In addition, business Web sites must be easy to scale in size (quickly add and subtract new pages) and update (change ads, change promotions, and so on). To make these two things happen, Web pages must have extra programming code added into them to make them dynamic. In the Web design world, a

dynamic Web page is automated through the use of programming languages that interact with online server scripts and databases. Additionally, the content of the Web site is typically separated from the HTML pages and is fed to each page from a database. By separating the "content layer" from the "presentation layer" as they say in Web design, the site becomes easier to manage and update.

Client-side and server-side programming languages

You can add two kinds of scripting to your Web page to make it more dynamic: client-side scripts and server-side scripts. You embed these scripts, written in programming languages such as Application Server Pages (ASP), Java Server Pages (JSP), Common Gateway Interface (CGI), Hypertext Preprocessor (PHP), ColdFusion (CFM), and Visual Basic Script (VBScript), right into the HTML page.

Client-side scripts are those that are self-contained right on the HTML page and do not rely on an external script or database in order to function. Here's an example of a simple client-side VBScript that greets a user in a small window that is separate from the Web page:

```
<HTML>
<TITLE>Sample of VBScript</TITLE>
This page has a tiny VBScript program that greets the user.
<SCRIPT Language=VBScript>
Msgbox "Howdy partner"
</SCRIPT>
</HTML>
```

Server-side scripts are written with languages like VBScript and PHP. Server-side scripts enable a Web page to open a connection to a database where certain information, such as product names, pictures, and prices, is stored. After the connection is established, the programming code can retrieve this data, and the HTML or CSS can format and place it on the page. The page can also collect information, such as a customer's registration info, and code on the page sends it back to the database.

For example, imagine an online store with hundreds of products. Building each product page by hand is simply not practical. Prices change on a daily basis, products change or are discontinued, and others are added to the mix. Building one Web page as a template to display each product is much more efficient. The HTML portion takes care of the page presentation, while the embedded programming retrieves the correct product image, description, and price from the online database. If all the product information is in a central database or content management system, you can make a change to a product's price once. Everywhere that price appears on the site is instantly updated.

Server-side includes

Server-side includes (SSI) enable you to populate a Web page with content on the fly as users download it. To work correctly, the Web page using SSI must be named with the `.shtm` or `.shtml` file extension rather than the usual `.htm` or `.html` extension. This extension tells the server to look for the SSI commands on the page, find the appropriate content on the server, and deliver that to the page. This tactic is useful for frequently used content items that occur on multiple pages throughout your site, such as headers, navigation systems, and footers.

The best part about using SSI is that if you ever need to make a change to one of the dynamically inserted content pieces, such as the header, you just need to make the change once. By changing the one header file, all the pages in the Web site that include it automatically update.

Although server-side includes sound like amazing timesavers from a site maintenance point of view, they do put a strain on the servers: Every time that a user comes to the site, each page has to access the server to retrieve some of its content. This can significantly slow down the site's performance and adversely affect the user experience.

Content management systems

A content management system (CMS) is used to organize, store, and manage all the text and images of your Web site. Through a front-end software application, you and your co-workers can go into the content database and make changes that you can then *push* live to the Web site.

Effectively, a CMS allows you to make changes to your Web site on an ongoing basis without ever having to edit HTML. A number of companies have created their own propriety CMS software, but there are also commercial providers like Vignette.

Baking your own cookies and shopping carts

If you'd like to utilize cookies on your Web site and build online stores but don't know how to begin, try WebAssist's WA Cookies and WA eCommerce Suite software at `www.webassist.com`. An easy-to-use interface walks you through the process of building and customizing your cookies so you don't have to write code. Ample help documentation and examples guide you through. WebAssist also provides a host of other Web productivity and design software — even software to build an eBay store — to give you a substantial leg up in building business and personal Web sites.

Personalized Web pages with cookies

I'm sure you've visited a Web site that knows who you are and immediately greets you by name, even without logging in. The magic sauce that makes this sort of personalization possible comprises little bits of code called cookies.

WEB SPEAK

In the Web design world, *cookies* are little pieces of data that a Web page leaves behind on your computer — sort of like a trail of data crumbs. As you use the site (for example, to order a product), the cookie stores information about your computer, your preferences, your name, and so on. Cookies can be one of two types:

✔ **Session:** This cookie is only a temporary resident on your computer and stores information as you move from page to page. As soon as you quit the browser, the cookie is deleted, and none of your preferences are recorded for the next time you visit.

✔ **Permanent:** This cookie is installed on your computer and keeps a running tab of data, storing your name, preferences, and other information so that they automatically appear on the Web site the next time you come back, even if you've restarted your machine.

Users can turn off the cookie feature in their browsers, so if you plan to use cookies, make sure that your Web pages can work without them. The user's experience shouldn't rely on cookies, just be enhanced by them.

Cookies can also pose a security risk for users. Because they store personal information like the user's name or Web site login information, cookies can be a target for other Web sites trying to get a user's personal information. Only the Web site that left the cookie on the computer is supposed to be able to retrieve data from it, but some folks have suspected that it's possible for other sites to hack into them. For more information on cookies, see Cookie Central (www.cookiecentral.com).

E-commerce shopping carts

To facilitate online shopping, designers and programmers have come up with a shopping cart metaphor that enables people to pick out things to buy and add them to their virtual shopping basket. Shopping carts are not tangible — you don't buy a particular piece of software and plug it into your Web site to create a shopping cart (unless you are using WebAssist's eCommerce Suite, shown in Figure 15-3). A shopping cart is more of a user interface design metaphor that enables people to pick out multiple things as they shop, review their items, make changes to their order, see how much everything costs, and then make purchases with their credit card.

Figure 15-3: WebAsisst's eCart, part of WA eCommerce Suite, allows you to step through the process of building an online store.

To orchestrate all these features, a series of Web pages, online server scripts, and databases work together. The Web pages use technologies such as ASP, JSP, and CGI written with programming languages such as Perl, ColdFusion, and VBScript to connect with online scripts and databases to process a customer's order and credit card information.

Part V
The Part of Tens

The 5th Wave — By Rich Tennant

RURAL WEB DESIGN

"What you want to do, is balance the image of the pick-up truck sittin' behind your home page, with a busted washing machine in the foreground."

In this part . . .

No *For Dummies* book would be complete without the requisite lists of ten that you can use as a handy reference. In this part, I compile convenient lists of all the important stuff you need to know — from tips to running your own Web design business to user interface techniques and HTML tricks.

If you read nothing else in this book, read this part: It gives you the quick basics to get you designing effective Web sites in no time. Pay special attention to Chapter 18, which concludes this book with a Top Ten list of the things that can go wrong in the Web development process so you can plan for these situations and know how to respond like a pro.

HOME   TRAINING

bike gear
aerobars
handlebars
brake levers
shifters
> bike trainers
wheels
tires & tubes
stems
forks
headsets
water bottles
food holders
cycle shoes
cranks

run
stuff

Ten Tips for Managing Your Web Design Business

In This Chapter

▶ Assembling an online and offline portfolio

▶ Talking about your work with clients and employers

▶ Assembling a proposal

▶ Determining your hourly consulting rate

▶ Understanding how Web agencies charge for work

▶ Managing a client's expectations

▶ Setting and enforcing a client's responsibilities

▶ Managing a Web project

▶ Hiring and managing subcontractors

My favorite saying that sums up almost any professional's life is "You go to work for a company for the illusion of security, and you go to work for yourself for the illusion of freedom." Nothing could be truer. When you start your own Web design business, you spend more time than you realize doing non-design-oriented stuff like assembling presentations, billing, collecting, and marketing your services. All your free time goes out the nearest open window.

On the flip side, when you work for a company — especially in the design services arena — you never know how the company will shrink and expand with the changing market forces and when you might be laid off. For these reasons, it's good to be proficient working in both settings: as an independent consultant working for yourself, and as a designer working with a team in a larger organization.

In this chapter, I offer tips on how to do the ten tasks that I've found most crucial for your Web design career, either when you're on your own or when you're working for a company.

Presenting Your Work

Whether you are going on a job interview to work at a company, presenting to an internal client, or presenting your work to your freelance clients, keep the following techniques in mind when assembling and presenting your portfolios.

Assembling a portfolio

Here are some tips for assembling your portfolio:

✔ **Build an online portfolio Web site.** Often, a client asks for a list of URLs to Web sites you've designed. Rather than just e-mailing a list of URLs to the client, assemble samples of your work in one nicely designed online portfolio site and e-mail just the URL of your portfolio site.

By making your own online portfolio site, you can also show work that's no longer live on the Web. Rather than providing a link to a nonexistent site, you can show images from the site and provide a little blurb about the project. In fact, I like to show images (scaled down to about one-quarter size) of the Web site and include a little paragraph that describes what I did, what design challenges I encountered, and how I solved those challenges. As shown in Figure 16-1, providing a little background on the project helps clients and employers better evaluate your designs.

✔ **Build an offline portfolio book.** In addition to your online portfolio Web site, you should assemble a book full of printed editions of your work. You can buy any one of a number of cool-looking portfolio books at your local art (not craft) store for about $80. These usually have black paper in a binderlike book, so you can take the pages out and rearrange them as needed.

You may be surprised to find out that a lot of employers at design agencies ask you to send your portfolio to them, rather than ask you to bring it in personally. If you aren't present, your book is your only representation, so you've got to make sure that it is polished, consistent, and professional.

Take screen shots of your work and print them at full size in full color on glossy paper. My feeling is that glossy paper gives you better color results than matte paper. You can decide whether to leave the browser interface in the screen shots, but whatever you do, do it consistently. Use a light spray adhesive, such as 3M Spray Mount, or double-sided removable mounting tape to adhere your prints onto the black paper of the book. Finally, make sure that your book is not too big and not too small — 14 by 17 inches is a good size to shoot for.

✔ **Put together a biography.** If you're an independent consultant, a client does not want to look at your résumé to get a sense of your qualifications. A client looks mainly at your portfolio of work. Still, providing a short paragraph that outlines your professional experience and accomplishments is

a good idea. Remember, after you sell a client on your services, the client has to turn around and sell you to the other people he or she works with. If the client can rattle off a few fun facts about you to grease the skids, his or her job is a little easier.

In addition, a bio is helpful to include at the top of your résumé when you're seeking a design job at a company. A bio is like an executive summary that sums up your experience and gives you a chance to sell yourself before the potential employer drills into the job-history listings.

Figure 16-1: This PDF "capabilities" presentation provides a case study on each page.

Taking screen shots

When building your on and offline portfolio components, at some point, you'll probably need to take screen shots of your online work. To take a screen shot, use either the computer's built-in capabilities or a special screen capture utility such as HyperSnap or Snapz Pro.

Mac users can press ⌘+Shift+3 to take a picture of the whole screen or ⌘+Shift+4 to draw a box around just a portion of the screen (with this latter option, the cursor changes to crosshairs, and you drag a marquee around the portion of the screen you want). The resulting screen shot is saved to your hard drive as `Picture 1`. (After it's on your hard drive as `Picture 1`, you can rename it to better remember what the screen shot represents.)

Windows users can press the Print Scrn key to capture an image of the active window. This key instructs the computer to take a screen shot and store it on the Clipboard. Next, open any graphics program such as Photoshop, Paint Shop Pro, or Fireworks, start a new file, and paste the screen shot from the Clipboard.

You can also download a screen capture utility (such as HyperSnap or Snapz Pro) from the Web. These utilities are better than the computer's built-in utilities because they capture mouse-down activities, such as selecting items in pop-up menus.

✔ **HyperSnap:** `www.hypersnap-dx.com/hsdx/`

✔ **Snapz Pro:** `www.ambrosiaSW.com/Products/SnapzPro.html`

Presenting your work

When you're ready to present your work, keep the following points in mind:

✔ **Have presence.** Presentation is half the battle. For the meeting itself, dress the part, have your material ready in all its forms — online and offline — and be bright, positive, and confident. Speak clearly and convincingly, make eye contact, and read and respond to the body language of the room. The worst thing you can do is come off nervous, soft-spoken, and unprepared.

✔ **Be positive and informative about your work.** The biggest mistake that new designers make is being too humble about their work (or even making excuses for — or berating — their work). As I say, "If it's good enough to show, it's good enough to support." Talk positively about your work. Discuss design challenges you may have faced and how you solved them. This approach shows the thinking behind your design treatments, giving them context and relevance so the client is better able to judge them.

✔ **Don't be married to your work.** Don't be upset if your work is not well received, or another designer's work is chosen instead. As soon as you produce a design candidate, remove yourself emotionally from it. Present it objectively and never defensively. It's bad form, and it's not going to advance the project. Discuss the feedback and, together with the clients, think through ways to change it more to their liking.

Developing a Proposal

When clients want work to be done, they prepare an *RFP* (Request For Proposal). An RFP is a document that outlines the goals and scope of the project so designers can better prepare a proposal. Often, however, the clients themselves are not sure what they want or what can be done, so the RFP is not as helpful as you may hope.

Oftentimes the RFP outlines clear, sometimes strict, procedures for submitting your proposal, telling you what to include, when to submit it, and what to expect after that. For those RFPs that are less specific, here's a list of things you should include in your RFP response:

✔ **Project summary.** Include a section that outlines the project, any specific ideas you have for the project, assumptions you have about the project, and any unique qualifications you and your team have for the project.

✔ **Project budget.** The most important aspect to include in your proposal is the bottom line: How much is this project going to cost the client? Provide an estimate for the project that covers the work assumptions and ideas you stated in the project summary and timeline. Clients often include a budget range in their RFP. If so, you can work backwards from the budget amount and scale the production effort to fit. When the client does not include such info, you must estimate what the project will cost based on the scope of the work and the schedule in your proposal.

When estimating, always add 20 to 25 percent more than you think that the project will cost. You can always impress the client by billing less, and if you end up needing the additional amount, you'll be thankful that you built in the extra padding. Regardless of how the project goes, asking the client for more money is always bad form unless the client has asked you to increase the scope of the project. Clients expect you to put enough time into the proposal to accurately predict your costs and profit margin.

✔ **Visual examples.** Clients usually respond better to visuals than a lot of text. Wherever you can, include diagrams and sample designs and deliverables.

✔ **Market and competitive analysis.** Depending on the nature of the project, it may be helpful to provide a market analysis section that shows competitive sites and discusses ways to differentiate the client's site. If you are developing a Web site for a commercial enterprise, such as an online store, it's helpful to do a little research into similar Web sites to make sure that the design you propose is competitive.

✔ **Your company background.** Include a section that provides an overview of your design agency or consultancy along with case studies of relevant projects you've completed. Also include a short bio on you and other key team members.

✔ **Outline of content and special features.** Create a basic outline of the content and features you propose for the Web site based on what the RFP states and the brainstorming you've done with the client. For example, if you think that an interactive timeline would be a great addition to the Web site, list it as a special feature and describe how it would work.

✔ **Sample navigation ideas.** Along with a list of content and features, you may even go so far as to suggest how you'd organize the interface. For example, you can outline a list of main categories and subcategories and even outline how the interface might work.

✔ **Production schedule.** Include a section that outlines the production schedule, complete with client sign-off points, your team's milestones, and the client's *deliverables* (tasks that the client is responsible for). A client sign-off point is when the client formally accepts the recent progress (by signing a document to that effect) and knows that they cannot ask for revisions without incurring additional costs. It's also important to determine upfront who has sign-off authority for the client. Also, many times, the client includes a desired due date for the project. In this case, you can work backwards from the date and scale the production effort accordingly. For example, if the client wants the project done in just one month, the scope of what you can accomplish is already limited.

The client's schedule is a very important element in your schedule. Client sign-off points are necessary because they enable you to close one phase of production and move on to the next, knowing that everything up to that point has been accepted.

Client deliverables are tasks the client is responsible for, such as providing you with the necessary images and content. Outlining a client's deliverables in writing is critical because it is the only leverage you have if the schedule slips due to the client's neglect. By listing both client and team deliverables, everyone agrees to their respective responsibilities upfront and avoids surprises later.

Winning the Bid

Writing proposals burns a lot of time — design firms and independent contractors all feel the same pain. Nonetheless, you must invest this time in order to even have a chance to win the project, so make your time count.

After spending a few years on the client side within a big company, I've seen a lot of proposals from the biggest Web design agencies in the country. The ones that won the bids all had these characteristics:

- ✔ **Demonstrated project relevance.** Proposals that discuss specific creative ideas and solutions for the project do better than generic proposals. By generic, I mean that some proposals only give background on the agency itself, its awards, its methodology (the same one I outline in Chapter 2), and then go into the project's timeline, schedule, and budget. Blah, blah, blah. Clients want to know how you would approach their project and hear some of your ideas.

- ✔ **Provided sample visuals.** Great designs that you've done for similar projects go a long way towards winning the bid. In fact, some clients even ask agencies to provide sample design directions for their particular project in the proposal. Such spec work is risky, but it's often the only way to win the bid in competitive situations. For the most part, however, sharing case studies of related work and pointing out relevant features create a powerful, and lasting, impression.

- ✔ **Touted solid team members.** Include a section in your proposal that provides a brief bio on each team member that would be working on the project. Clients want to know that they'll have an all-star team dedicated to their project. And, if you win the bid, make sure those team members are in fact on the project! I can think of one large project where the agency swapped in junior team members. The project fell apart, created a huge amount of friction, and opened the door for legal action.

Knowing What to Charge as an Independent Consultant

Knowing what to charge is always hard. You can use any one of various formulas to arrive at an hourly rate that takes into consideration your annual expenses, profit margin, and salary, but you can arrive at this number through simple common sense:

- ✔ **Ask around to find out what other designers are charging.** This information can give you a good reality check as well as a range of prices. You may find that freelance designers in your area are charging between $50 and $150 an hour.

- ✔ **Be honest about your level of skill and experience.** If you've been around for a while and you have a range of high-profile sites in your portfolio, you probably know exactly what you should charge for your freelance services, and it's probably towards the top end of the range. If you're new to Web design but are an old hat at print design, your fee may be somewhere in the middle.

- ✔ **Estimate your salary and expenses.** As an independent consultant, you have to calculate what it costs you to run your business each month, how much you want to make, how much time you can honestly bill in each month, and taxes. (As an independent freelancer, you get taxed more than a full-time employee does, but you can also deduct a lot more. Ask your tax professional for guidance here.)

- ✔ **Think of all the things you need to buy in order to run your business.** Electricity, office supplies, computers, software, fonts, an Internet connection, trade magazines and organizations, and so on all add up. Think of a monthly budget for all these things, and then think of what you'd like to make on top of that. For example, if it costs you $4,000 a month to run your home office and you want to clear $6,000 a month, you have to figure out how to make $10,000 a month.

- ✔ **Figure out how many hours you must bill each month.** *Billable time* is all the time you actually spend doing client work. Ideally, this is at least half of your time; but more often than not, checking e-mail, writing proposals, and performing other activities cut drastically into your available time.

    If you assume that each year has 50 workweeks (leaving two for vacation), you have 4.1 weeks in a month. At 50 percent billable time, that leaves 83.3 hours of billable time. To make $10,000 a month, your hourly rate needs to be $120.00.

Incidentally, I've noticed that a freelance hourly rate oddly corresponds to an annual salary. Notice that $10,000 a month is $120,000 per year, and the hourly rate is $120.00. The same phenomenon occurs in the workplace. For instance, a high-level designer who is paid $150,000 a year in an agency can probably charge about $150.00 an hour for freelance work. Similarly, a junior designer who makes about $50,000 a year for an agency can charge about $50.00 an hour for consulting work.

When you bid on a project, use your hourly rate to come up with an estimate of what the project will cost, but ultimately quote clients a *fixed bid* (a single flat fee). It's better to charge clients a flat per-project fee than to charge them hourly. Firstly, most clients expect a fixed bid so that they know exactly what the project will cost.

Another reason to charge a fixed bid is that if you work fast and zero in on the design quickly, you are paid for the *value* of your work, not just the few hours it took you to knock it out. Make sure that you're paid for using your brain, not your hands!

The hardest part of charging a flat rate is accurately estimating the amount of work and the time it will take you to complete the project. Spend time thinking through each step of the proposal, gauging the work, estimating how long each step will take, and putting a dollar figure next to it. Include any subcontractor's estimates, and then add up all the steps. Add 20 to 25 percent to the budget for good measure.

How Agencies Charge

Web design agencies use formulas similar to those used by freelancers for calculating their internal hourly rates. Generally, agencies have different billing rates for each level of designer, from production artists on up to creative directors. The prices that Web design agencies charge clients for these designers, however, are a lot higher than what the agency pays their designers. Although such agencies may pay a junior designer $50,000 a year, they may bill that person out at $100 per hour when they are calculating the costs of a project.

Although such a high price sounds heavily bloated, you must consider that agencies have a lot of overhead expenses to cover. In addition to the normal lease payments and supply expenses, agencies have a lot of non-billable, but valuable, support people, such as administrative assistants and accountants. The billable people in the organization pay for the non-billable people.

The larger the organization gets, the more expensive it becomes to keep the ship afloat. For this reason, typically each Web design firm has what is called a *minimum size of engagement*. If a client called a big design firm for a project that had a budget of only $50,000, that firm would probably refer the client to a smaller agency. Big agencies simply cannot afford to take projects unless they meet a certain budget range.

Because of an agency's minimum engagement budget rule, the independent consultants and the smaller design houses play an important role in taking on the multitude of smaller projects with budgets from $5,000 to $100,000.

Managing a Client's Expectations

Above all else, setting and managing a client's expectations before and during a project are among the most important tasks you have as a designer. No two people ever hear or see the same thing. Even when you're explaining a project to a client, the client may be thinking one thing when you mean something else. For example, if you require content from the client, make sure that the client knows when and how to deliver it to you. Also make sure that the full range of services you do and don't provide is clearly outlined in the proposal.

WARNING! If you have nothing in writing, you'll have a difficult time describing why a certain feature was left out of the project or why the schedule slipped. Whatever goes wrong is always your fault. When assembling the proposal, the best way to protect yourself and ward off any potential conflicts with the client is to be very clear about due dates and what content will be included in each deliverable.

Setting Client Responsibilities for the Project

In the project proposal, one of the most important elements to list is the client's responsibilities. Make sure to discuss these seriously with the client at the beginning of the project. Clients must understand that the project will stop in its tracks if they do not meet their deadlines for delivering content or approvals to you.

For some reason, clients tend to think that they don't have to do a thing after they sign the contract. They don't realize that you can't do your job without getting content from them. For example, when you build the product section of the Web site, you'll need photos and information for each of the client's products. Unless taking new photos and writing new copy are part of your proposal, the client must provide this material in a timely manner.

TIP Just to cover yourself, pad extra time into the schedule for each client deliverable without telling them. This way, even if the client slips, you won't, and you can better schedule your team's resources and time.

Getting Clients to Sign Off on Key Milestones

In addition to feeding content and materials to you during the project, clients must also sign off on various key milestones along the way. By getting clients to sign off on key steps in the project, such as the site map, you protect yourself from any future arguments over the scope and quality of the project. At the beginning of the project, it's also very important to establish who has sign-off authority for the client. In some cases, a different person in the client's organization may sign off for different parts of the project.

For example, imagine that you are halfway into HTML production and your client's CEO balks, refusing to pay unless you include an interactive company history page. If you have a signed site map with no history page on it, you can clearly state that the page was never part of the scope, and you have so-and-so's authorized signature to prove it.

Aside from protecting you from clients' tendencies to change their minds and demand new features, a sign-off policy also protects the clients. They have signed documents that assure them and their managers that they will get what they're paying for.

While most clients are reasonable, a handful out there (think large entertainment companies) are known for delaying feedback, noodling designs ad nauseam (because they can't make a decision, change their minds, or cannot be pleased by any design that is not their own), delaying projects midstream, the list goes on. If you suspect you are going to be dealing with one of these companies, be very clear about what their fixed-bid proposal gets them and when your hourly rates kick in. For example, your proposal might include three design directions of a home page and a subpage, and two rounds of revisions on one selected design. Any noodling thereafter will incur hourly rates at *X* dollars an hour.

Managing the Web Project Workflow

When you're knee-deep in a project, you have the internal challenge of managing people on your team to get the job done. For the most part, if you're working within a larger organization, project managers or producers manage the client and the team members to ensure that all the resources (content, people, and so on) are in place and to ensure that the project stays on track. This management model enables you, the designer, to focus on what you do well.

If you are on your own, managing the client and the project schedule and doing all the design and production work can keep you working around the clock.

If you're on your own, you might pick one thing that you do well, and that you enjoy, and offer only that service. Doing all aspects of Web site design from creative to programming is difficult enough; *managing* the whole process by yourself is even more difficult. If you do only one thing (such as information design, Flash animation, or Web creative direction), you must manage only one deliverable. When you are a specialist, marketing your services is easier than if you're a generalist. You can market directly to clients, to design agencies, and even to other freelance designers and producers who need your piece of the puzzle on their projects.

Hiring and Managing Subcontractors

As your Web design consulting business expands, you may find that you're getting more work than you can handle. If you're like me, you don't like to say no to more work! One way to handle your growing business is to find able-bodied freelancers like you that you can rely on for expanded project needs — and then manage them as the project's producer.

For example, one client may want you to develop a series of four design directions at the very same time that another client needs work done. In cases like this, you may subcontract another designer to help you create a few of the design directions so you can get everything done on time.

When hiring subcontractors, their rates may often be close to your own rates. Plan for your subcontractors by getting a fixed-bid estimate from them that you can roll into your overall project budget.

Marking up their services by about 15 percent is acceptable. After all, you need to be compensated for the time spent managing them.

WARNING!

Look at your initial client contract to see whether hiring subcontractors raises any legal issues. Often, a client simply signs the project proposal and no other legal agreement. In such cases, you are free to hire and manage subcontractors as needed. If the client, however, asks you to sign a work-for-hire agreement along with the proposal, the agreement sometimes forbids any subcontracting activity.

As for signing any agreements with your subcontractors, you too may keep your own standard work-for-hire agreement on hand for them to sign — especially for larger projects. For small projects, however, the subcontractor usually just puts a mini-proposal together that shows the work to be done, the price, and the schedule, and that's enough to go on.

At the end of the project, the subcontractor sends you an invoice. Keep these invoices in a safe place and make sure that they include the following information (you need it for tax purposes at the end of the year):

- First and last name
- Address
- Phone number
- Social Security number or Federal ID number

Like sending out a cynical version of holiday cards, at the end of the year (in the United States) you must send out 1099 forms to every subcontractor to whom you've paid more than $600 throughout the tax year. Total up the amount you've paid them and fill out one 1099 form for each person. You can find these forms at any office supply store or at your local post office.

You must send the 1099 forms out in the mail by the end of January, or you may not be able to claim the invoices as expenses on your own taxes. Again, talk to your tax professional.

Ten Information and Interaction Design Tips

*A*s a Web designer, you must be familiar with strategies that enhance the usability of a site. The way you structure information and design navigation systems to get around are the crux of good site design. Once the bones (as I like to say) are worked out, the next layer is the page-level interaction and visual design of the site.

In one convenient chapter, I've consolidated the top ten information, interaction, and visual design strategies that you can keep on hand the next time you embark on a Web site project.

Use Only Five to Seven Main Categories

Five and seven are magic numbers in life because remembering a list of five to seven things is easy. Any more than that and our brains lose track. Maybe it's because we have five fingers on each hand. We can mentally attach one item to each digit, and if we've had enough coffee, we can remember a few more items on the next hand.

You may think that I'm joking, but I've heard from psychologists-turned-interface-designers that the five finger phenomenon actually has merit. In Web design, many user interface designers suggest that keeping your list of categories down to just five to seven is best. This strategy keeps users from feeling overwhelmed in your site.

TIP

The problem with the five-to-seven rule, however, is that most modern Web sites have a lot more going on, and honing the site down to just five or so areas is difficult. In such cases, I've found that breaking the navigation into sets is helpful: a primary, secondary, and tertiary set if needed.

Each set can contain five to seven items. Take a look at the diagram in Figure 17-1. In this design, the top navigation area has five links to the main categories of the site. The side navigation area has another five links, and the bottom footer has four links. The content in the middle may have a few links too, but they are shortcuts to sections otherwise accessible in the three navigation sets.

Keep in mind that this rule applies to site global navigation and not to links that are content like a list of resource links or a list of anchor links.

Figure 17-1: Each of the three functional areas contains five to seven links.

Develop Wireframes for Each Unique Page Layout

One of the most important steps of the Web design process, and one that you should never skip even for small scale sites, is the development of wireframes. Like blueprints for a house, wireframes are diagram-like drawings that articulate your thoughts for laying out content on the page and figuring out the interaction required to get through the content.

You do not have to do a wireframe for each page of the site — only the unique layouts and pages with intricate interaction, such as the pages in a store checkout process.

Wireframes are your chance to work out all the navigation nuances of the site, accommodate all the content the site will have, and get a rough idea of how everything will fit on each page before you begin developing visual designs.

In fact, I like to have the first pass of wireframes complete before I begin exploring visual design options. Without knowledge of at least the rough navigation scheme and content elements, it's difficult to develop a visual design that works for the site. Exploring designs too early in the process puts you at risk of developing something that the client loves but is just not practical and requires significant reworking. Time and budget wise, waiting to do the visual design is most efficient.

Always Label Your Buttons and Icons

As idiot-proof as you think your icon or illustration may be, I've found that you can never rely on pictures alone to tell users what a button does. Unless you're designing a print function or other commonly recognized task, adding a simple text label to a button or icon is a good idea. After all, no picture could reliably represent the Product Catalog section of a site.

By the time you illustrate an icon detailed enough to give users a good idea of what the section is, you have a picture worth framing. You may as well save some space and add a simple text label. I'm not saying to not use icons; they can add a lot of design flavor to a site. Just be sure to supplement them with a label for clarity's sake.

Mind the Download Time

To maximize the user experience of your site, always keep in mind the connection speed of your end users and the ease of which they can access key areas of your site. For example, avoid long all-Flash intro sequences that must play through before users can make a navigation choice. Even if your Flash intro has a Skip Intro button, you force users to download the Flash

movie to the point where they can even click the Skip button. Only then can they finally access the global navigation. Such a two-step process that can also involve a long wait is annoying for your repeat visitors.

A better plan is to load the global navigation immediately along with the Flash segment. That way people can bypass the Flash and get directly to the content at hand. Another strategy is to include a Cover page that allows people to launch the Flash-enabled site (which usually opens in a new window with a custom size).

Provide "You Are Here" Feedback

The navigation system that you design should not only provide access to all the main functional areas of the site, but also give users some sense as to where they are in the Web site. Like a mouse able to see the maze from an aerial view, your navigation system should provide the same sense of orientation and visually show people the size and scope of the site.

In Figure 17-2, a small map of the whole site immediately gives users a view of the site and a quick means to navigate through it. When a user rolls his mouse pointer over each little icon, the icon is highlighted, and a label that tells the user what it represents appears. A bread crumb trail provides continual feedback of where you are in the site.

Figure 17-2: This navigation system shows a miniature view of the entire site.

Make It Easy to Get Back Home

One of the functional items people forget to include most often in Web design is a link back to the home page. People drill down in a site, find the info they need, and then suddenly realize that the only way to get back to the home page is by reentering the URL.

Often, the company's logo at the top of the page is a secret passage back to the home page. Seasoned Web surfers usually try to click the logo to get back home, but to a novice user, the logo just looks like a logo.

TIP

The best course of action is to include a dedicated link to the home page as part of your standard set of navigational buttons — usually in the secondary or tertiary navigation set. Figures 17-3 and 17-4 show different ways sites often treat home links.

A home page button is included in the main navigation.

Figure 17-3: You can count the home page as one of the main sections of your Web site.

Figure 17-4: Most often, the home link is found in a secondary or tertiary navigation set.

Visually Differentiate Clickable and Nonclickable Things

Although reusing graphics wherever you can to take advantage of the browser's caching ability is tempting, don't use the same graphic as a button on one page and as a decorative headline on another. For example, if you use an icon as a button leading to the About section of the site, don't use the same icon merely as decoration for the headline on that page. Otherwise, people will still think that it's a button. Give the icon a slightly different visual treatment so people know that its function has changed.

REMEMBER

You should always treat the visual design of clickable things differently than the design of headlines, images, and other nonclickable things. Design interactive elements to look like their function. For example, linked text should always have a unique color, or be bold or underlined to separate it from normal body text. It's the same for linked headlines versus informational text. Buttons should also have a unique and consistent treatment throughout the site.

Develop a style guide for all design elements in your site — for links, text, a few levels of headlines, buttons, table elements, bullets, and other elements. After users figure out your system, your site becomes easier to navigate.

"One of These Buttons Is Not Like the Others"

Continuing along the same lines as #6 in this list ("Make It Easy to Get Back Home") is consistency. If you've ever watched *Sesame Street,* you may remember the famous skit: "One of these things is not like the others." In this scenario, kids are taught to weed out the objects that don't fit with the rest of the group. This is an excellent analogy for user interface design.

Consistency applies not only to the visual treatment of your navigation sets, but also to their placement on the page, and logical grouping. Your primary, secondary, and tertiary navigation sets should always be located in the same relative location on the page. In addition, the links you include in each navigation set should be similar in function and priority. For example, if you have a few tool-like functions like "view map" and "view calendar," these should be grouped together, visually similar, and separate from other navigation sets.

The best interfaces are those that remain consistent throughout a site. The user becomes comfortable with the buttons and remembers where to find them. The interface becomes almost transparent, and the user can focus on the content of the page.

WARNING!

If you change the design or placement of a button from one page to the next, users may not find it, or they may think that the button has an entirely different function — even if it has the same text label. They may ignore it and continue to scan the page, searching for the button they just clicked.

Tread Lightly with Real-Life Metaphors

Sometimes clients want the interactive model for their Web site to be inspired by an object or a place, or modeled after an experience like watching a movie or playing a video game. For example, a client for a children's site might want to use a 3D clubhouse for the main interface.

Although real-life metaphors provide interesting ways to think about the design of a site, they can also impose a lot of design constraints on you. For instance, as in Figure 17-5, if you want the interface to look like a digital camera with chrome buttons surrounding a central screen, you suddenly have a small viewing space for all your content.

TIP

Instead of taking metaphors too literally, start the creative process with a theme, concept, or story. For example, the theme "march to your own beat" may inspire interesting visuals, a customizable soundtrack for the site, and a unique twist on the copy you write throughout the site — including names for the navigation. In addition, a theme can naturally extend to other supporting media such as brochures, HTML e-mails, ads, and other creative formats.

Figure 17-5: This interface looks like a digital camera and leaves little room in the interior window.

Use Color-Coding Sparingly

Another way to orient people in your Web site is to color-code each of the main sections. Color-coding is most useful in cases where the site has just a few sections, but each are deep in content. For example, a corporate site may have a branded online store that has a different color scheme than the main site. Or a conference Web site may have a few different tracks that you can color-code so prospective attendees know they are looking at the right one when thumbing through the session descriptions.

WARNING!

If you color-code a Web site, be sure to choose colors that work well together and have similar light and dark values. For instance, don't choose a set of five dark dingy colors and one bright yellow color. If you do, you won't be able to apply the same design strategy to each color-coded section. The bright yellow one needs dark text while the dingy dark colors need light text.

In Figure 17-6, the difference between dark and bright colors on a white background is apparent.

Figure 17-6: The first colors have similar luminance values; the fifth hardly shows up.

REMEMBER

A color-coded system works well only if you have a few main categories. If you have too many categories, you have difficulty choosing a set of colors that work well together and are still different enough from one another to make color-coding meaningful.

18

Ten Things That Can Go Wrong

In This Chapter

▶ Managing project scope creep

▶ Making time for a site map

▶ Presenting design options that you can live with

▶ Including user tests in your proposals

▶ Keeping technology tinkering under control

▶ Planning for international localization

▶ Managing database-driven sites

▶ Marketing your site

*Y*ou may know how to initiate a project with a client, build a site map, create design directions, and make technology choices, but you must know one more thing to be a successful Web designer: contingency planning. Anticipating the worst that can happen in the course of a project and planning how to deal with it are the last steps towards becoming a full-fledged Web designer. In this chapter, I list the top ten things that can go wrong, why they happen, and how you can respond to keep a project on track.

"Can We Add Just One More Thing?"

When clients first come to you, they often don't know what is possible with Web technologies. They don't know what you can and can't do or how much anything really costs, so they don't ask for it in their Request For Proposal (RFP). After a project is underway, however, and clients start to see the site take shape, their eyes tend to grow wider and wider with all the cool possibilities.

WARNING!

As clients become familiar with the Web development process, they may often ask you to throw in an extra Flash movie here, a personalized greeting there, and all manner of extra features to liven up the site. In Web design circles, these little additions to the project are called *scope creep.* If you give in to these little client requests, the scope of the project can slowly creep upwards until you are basically working for free — or worse, going into debt!

Identifying and resisting scope creep whenever it happens are critical. Aside from loosing your shirt financially, scope creep causes two other huge problems:

REMEMBER

- ✔ **The ripple effect.** Although a change may seem small at first, you must look at how it affects production of the rest of the site and the ongoing maintenance of the site. Sometimes just by changing one little thing, you break something elsewhere in the site because it was never planned for in the first place. Or, you cause unforeseen technical or customer support issues in the future. For example, adding a simple survey requires a live human on the client's end to assess the incoming data.

- ✔ **Production inefficiencies.** Because scope creep can come at any time in the Web design process, you cannot implement the new feature without causing production inefficiencies. Adding a new feature midstream causes the team to stop what it's doing, redo tasks already completed, and refocus on the addition. Start-to-finish, midstream changes take more time to implement than they would if they were in the initial plan.

Still, scope creep happens and, because you're the customer service type, you find it tough to say no. Here's how to gracefully manage scope creep:

- ✔ **Get everything in writing.** Before you begin work on a project, make sure that your proposal clearly states the scope of the project — what you're including, what the client must provide, and what the project does not include. Also make sure that your site map is detailed enough to show how all the proposed content works together.

  If everything is in writing, no one can question what was included in the original deal.

- ✔ **Share the budget ramifications.** When a client asks you to insert a little something in the site, say, "Sure, I'd be glad to! Let me come back to you with what that would cost and how much time it would add to the project."

  Need I continue? Sharing the realities of an expanded budget and time schedule often quickly turns a client around. If your client decides to move forward with the addition, at least you and the client have clear expectations about how it impacts the project.

"We Don't Have Time for a Site Map."

Often, clients come to designers at the very last moment and ask for an entire site designed and delivered within a ridiculous timeframe. The client insists on a crazy schedule, usually to try to meet an important conference or meeting on a hard date.

In situations like these, you may be tempted to dispense with the proposal, the project plan, and the site map so that you can dig right into the design directions. The problem with rushing in, however, is that neither you nor the client has a road map to guide you. Without a road map, you have no idea what to include in the design directions — you plod in the dark, wasting time as the deadline slowly ticks closer.

Even worse, when you dispense with the planning phase, you set yourself up for disaster in terms of client expectations. The client may be thinking one thing, and you may be thinking another. Midway through the project, the client may not see what he expected to see, and panic follows.

Ironically, the most time-efficient way to proceed is to invest a little time up front creating a proposal that clearly outlines the content and goals of the site and to follow that with a site map that shows how you plan to arrange the content. You can do both these things in just one to two days of working closely with the client. With a clear plan in hand, you can knock out the design directions and produce a site that the client loves with plenty of time to spare.

"The Clients Want THAT Design?"

When you're presenting design directions to clients, you may be tempted to shower them with a ton of options to choose from. Offering them lots of options gives them the impression that you've spent a lot of time thinking about their project, and it gives them a lot of ideas to consider.

The problem with this logic, however, is that it's difficult to come up with more than four to six directions that are distinctive and that you *like*. You always have a favorite and a least favorite design. For some reason unknown in the cosmos, clients have a knack for falling in love with your least favorite design. Therefore, never present a design that you can't live with.

To assemble a good group of designs to present, have a few different designers each come up with one or two designs. This way, you're sure to get an assortment of unique designs. From these, choose the top three to six designs that you feel good about presenting to the client. (Generally, three design directions are plenty!)

If you are an independent consultant and don't have a staff of designers, try finding a few like-minded independent designers to help you develop design directions for each project.

"Who Needs Usability Testing When You Have Me?"

You may laugh, but I've heard the sentiment, "Who needs to test when you have me?" expressed by more than one Web designer. Often, designers find that they simply don't have enough time or money to organize a formal user test for a new Web site — many times because the designers never planned for testing in the first place. All too often, testing is considered an unnecessary expense in the budget — even by the clients.

For large projects, a client can end up spending a lot of money on a project that simply doesn't work. Without testing, no one knows about problems until the site is live on the Web and the negative customer feedback starts rolling in. After a user has a bad experience at a site, a second visit is not likely. Testing is crucial.

For large enterprise sites, planning and budgeting for formal user tests are imperative. For small- to medium-sized sites, you can still plan for user testing, but you don't have to go all out with the formal, expensive, time-consuming procedures. Organizing small, informal testing intervals along the way using friends, colleagues, and even the clients themselves is better than nothing at all.

"But I'm Sure I Can Make This New Technology Work!"

As you wade knee-deep into production, the programming folks working around the clock can very easily lose track of the time schedule. Most programmers and HTML people that I've met love the challenge of solving problems and doing what others say can't be done.

The relentless pursuit of solving problems and adding cooler features (a.k.a. _gold plating_ in the industry), however, can quickly become a drain on the project's schedule. The project manager must keep a close eye to ensure that the

technology team stays on track and doesn't spend more than the allotted time on any one technical issue.

Tinkering is not limited to just the technology team: Designers are also known to push pixels around for hours until they get the perfect design. To keep a project on schedule, the project manager must stay on top of the milestones and where the team lies in the process.

"We're Planning for an International Audience?"

WEB SPEAK

Although most sites on the Web are in English, a growing number of companies are localizing their sites to cater to the needs of the international marketplace. *Localizing* entails translating the Web site into new languages and, in many cases, hosting the site in countries abroad.

If the Web is a global medium, why would you need to host a site in another country? If you've ever tried loading a Web site from across the world, you've probably noticed that the performance is pretty bad. A site hosted on a far-away server may be lean in terms of file size, but its sheer distance from you makes it load slowly. If you look at it from the European perspective of accessing sites hosted in the United States, you realize that a lot of European customers are experiencing far slower service. Asian customers in China have it worse. All inbound sites are filtered through "the great firewall of China" to ensure only culturally appropriate content gets through.

Along with hosting a site abroad for better performance, you may also consider translating the site into a number of languages. I've found that up to 50 percent of a Web company's business can come from overseas. The problem, however, is that this 50 percent is distributed across six or so languages, from German to Japanese.

TIP

When designing a Web site (in English), the guideline is to allow 30 percent more space in both the horizontal and vertical direction for text to accommodate the longer word and sentence structures of other languages like German and the height of Asian characters. Take a look at Figures 18-1 and 18-2. These figures show the same page from an online course in both English and German. A line-by-line comparison from the headline to the bulleted list shows how much more space the German language takes up on the page. If you are not careful in your initial planning, translating a site into another language may significantly alter the page layout.

Figure 18-1: A page from a training course in English.

Figure 18-2: The same page in German: Each line of text is longer.

"The Design Needs to Work on Windows?"

Most designers and creative types work on Macs, and most tech-heads work on PCs. Although most graphic design software tools these days look and function the same on both the Mac and Windows, the creative community's long-standing tradition to use a Mac still exists.

Because most people creating Web graphics use a Mac, a lot of graphics look great on a Mac but not on a PC. The two biggest design factors when designing cross-platform Web graphics on a Mac are the following:

✔ **Gamma differences.** Macs and PCs have different monitor brightness and color display settings. A Mac's display is much lighter and less saturated than a PC's. Therefore, graphics with dark colors and subtle color variations often appear solid black when viewed on a PC.

If you develop Web graphics on a Mac, always test your graphics on a PC before you include them on the site. Also, some software programs, such as Fireworks, have a Windows gamma preview feature that enables you to see how your work looks on a PC.

The reverse is true for those of you renegade designers building graphics on a PC. Always test your graphics on a Mac, or use a graphic program's gamma preview function to make sure that your graphics look right on both platforms.

✔ **Font size differences.** When you mock-up a Web page on a Mac — complete with dummy text that you'll ultimately replace with HTML text — the text looks smaller than you intended when viewed on a PC. This is because PCs usually have a 96 dot per inch (dpi) display, which is a finer dot size than the Mac's 72 dpi.

For mock-up purposes, try bumping up your font sizes by one point. For instance, instead of 10 point Times for the body text, use 11 point. If you're building graphical headlines, try making them slightly larger than you think they should be. Finally, always take a peek at your work on a PC before you send it off to the production team.

"Uh . . . It Needs to Work on a Mac?"

Because programmers and HTML slingers mainly work on PCs, they can easily assemble Web pages that look perfectly fine when viewed on PC browsers but are a mess on the Mac. Internet Explorer and Mozilla Firefox on a Mac and on a PC are completely different browsers. Usually, the Mac versions are a little behind in their capabilities.

Although Mac users comprise a much smaller segment of the Web-viewing public, they are still an important audience segment. When designing a site, your target audience's technological profile is an important factor to determine and plan for.

Before building a Web site, you and the client must decide which browsers and what versions of them the site should support. You should also determine which connection speed you will optimize the site for and which plug-ins you'll require. Then, as you get into production, make sure the development team has access to the target technology set up for the purposes of testing its work.

"We'll Just Make the Whole Thing Database-Driven."

If you read about all the cool technologies you can use to automate and scale a Web site that I discuss in Chapter 15, you may be tempted to make the whole site reliant on a database. After all, the database can populate each page of the site on the fly, so all you have to do is create a few template pages and voilà: an instant, automated Web site. To update the site's content, you need only to make changes to the online database, and presto, the whole site updates instantly.

Although such programming technologies and databases streamline the creation and maintenance of a Web site, they are not without their own special requirements:

✔ **A dedicated team.** To keep the technologies and databases kicking, the client needs a serious team of propeller-heads on staff around the clock to make sure nothing goes wrong. If the database gets corrupted (which happens more often than you might think) or if the database server goes down, the whole site goes down in flames, and your client ends up on the front page of the local newspaper (which happened to one famous auction site while I worked there).

✔ **Souped-up servers.** If a good portion of the site relies on database technologies, you're putting a strain on the servers and increasing the chances that they will give out. Remember, each time someone visits a page, that visitor accesses the database. With hundreds of visits each day to multiple pages, you're looking at a lot of server activity! Your server setup needs to be robust enough to handle the incoming load without toppling over.

If a significant portion of the Web site relies on database technologies, make sure that the client has a *hot* backup system in place. This is a redundant server with a copy of the database (backed up every 30 minutes or so) that can come online in a pinch if the main server goes down.

Another tactic is to have a different server housing each major database. This configuration mitigates the strain on each individual server and helps to isolate problems.

"If We Build It, They Will Come."

What worked in the movie *Field of Dreams* won't work for a Web site. With so many Web sites on the Internet, getting people's attention and drawing them to your site is very difficult. You cannot launch a site anymore and expect people to just find it or even care about your offerings. That's why you and the client must build a solid marketing plan and start executing it even before the site is live on the Web.

The best marketing plan for a Web site is one that involves a combination of traditional offline marketing and public relations techniques with online tactics. By offline, I mean every advertising medium that is not the Web: radio, trade shows, direct mail, billboards, print ads, and TV commercials. Online techniques can be much less expensive or free and just as effective. Some ideas include

- ✔ **Swapping banner ads or purchasing banner ad space.** Of course you can always purchase banner space on sites, but you can also be creative and try to work out relationships with other sites. Identify a number of Web sites that have a similar clientele and offer to swap ad banners. Often, to accommodate the disparity between the amount of traffic between your site and your partner's, you can work out additional forms of compensation. For example, if the partner's site has heavier traffic, you could sweeten the deal by offering a special discount on your products to that company's visitors.

- ✔ **Co-promoting with other companies.** Another effective marketing tactic is to partner with other online companies with customers who can benefit from your offerings. For example, if your site sells custom reading glasses, try partnering with a book club site and running a promotion such as, "Sign up today for the book club and receive 10% off ACME custom reading glasses."

  With this approach, you can have a much greater presence on the partner's Web site than with a banner ad, and your partner may even actively promote you in its online and offline marketing campaigns.

✔ **Leveraging search engines.** Search engine optimization, or *SEO* as it's called, is an inexpensive way to help promote your site. When people use a search engine such as `www.google.com` to look for a Web site, they enter a series of keywords and phrases such as "horse, sale, dressage, warmblood" into a search field. To find Web sites that match these keywords and phrases, search engines look through not only the text of Web pages, but also the page's title, and `<meta>` tags contained in their HTML code. When you build a site, make sure that such text can be found by search engines.

Index

• Q •

• R •

• S •

• W •