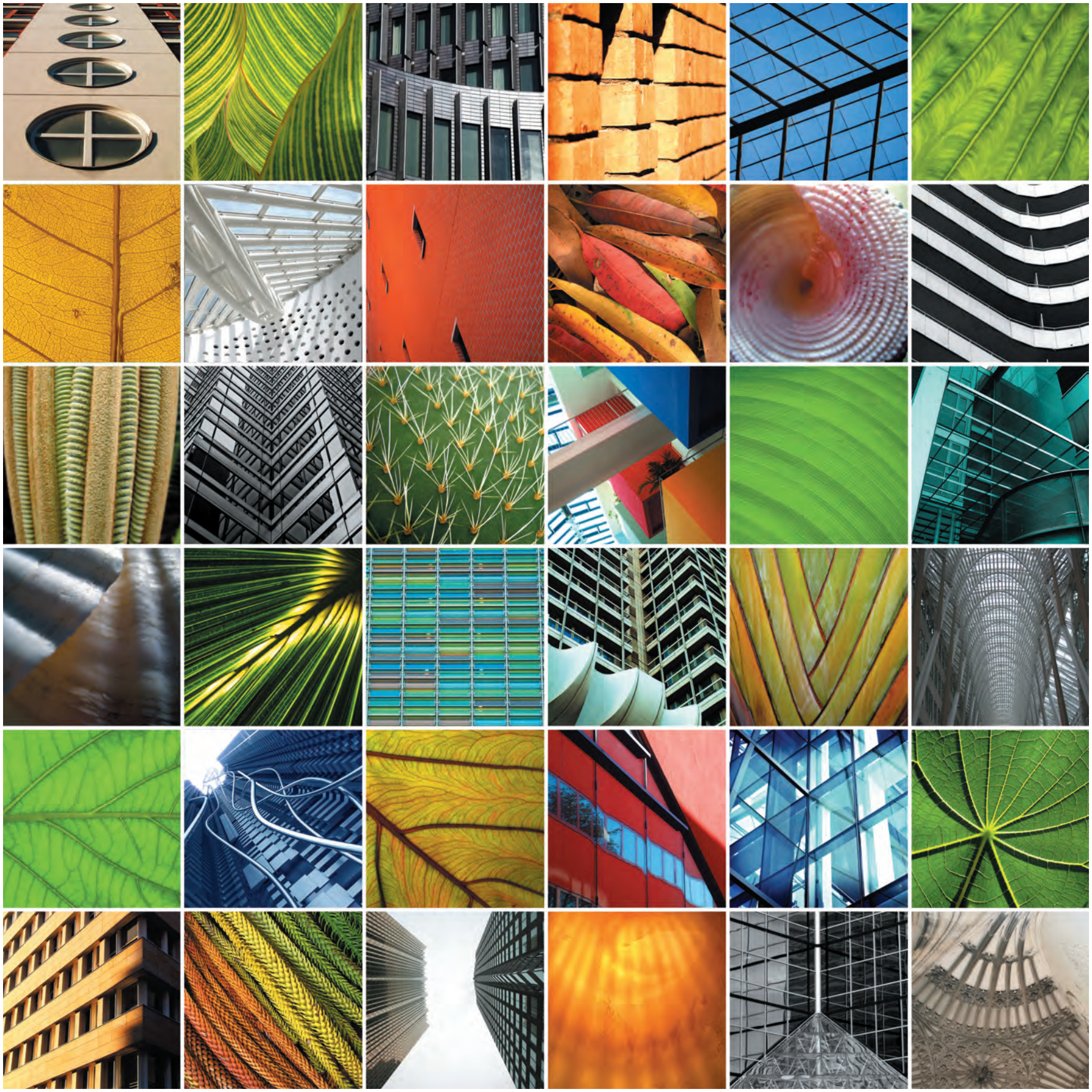


1 Discovery

Think differently about the Web design process.

See how to improve the design workflow.

Look forward to exciting possibilities.

Introducing Transcendent CSS

Transcendent CSS is more than a plea to use the latest, coolest CSS. It's a quest to use the lessons you're learning in CSS as a means to becoming the finest artist and designer you can be. Transcendent CSS asks you to embrace the new rather than the old and to stimulate new ways to find inspiration, create more agile and appropriate workflows for Web design, and encourage yourself to constantly learn more about both the design and the technical issues with which you work.

Which tools do you need to get started?

Which tools do you need to adopt the Transcendent CSS approach and to work along with the principles explained in this book? You don't need anything more than you are probably using already. Don't worry, you won't need a spanner or a monkey wrench. You won't even need special software or new server configurations.

This book is not aimed at beginners; I assume you already have a good, working knowledge of XHTML (eXtensible HTML) markup and CSS and you understand the core concepts of Web standards. If you are still at the stage of using tables for layout, this book won't teach you about the basics of selectors or common CSS properties; many other fantastic references are already available that will do just that.

But if you are a newcomer to CSS, I hope you will find the concepts and examples found in this book inspiring and that you will want to grab hold of the handlebars and learn as much as you can about CSS. No matter how long you have been working with CSS, you'll find new places to go and new things to learn.

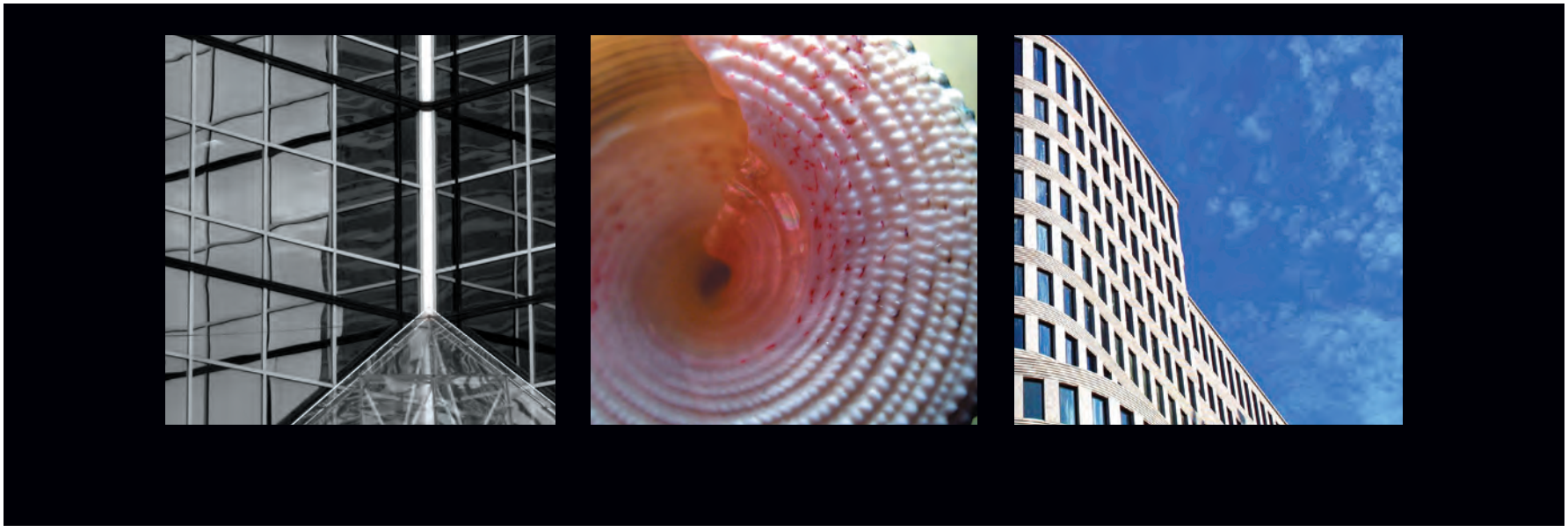
This book also assumes you have an open mind. Although not everything discussed will be 100 percent relevant to you or the work you're doing for your organization, studio, or clients, I encourage you to take on new ideas. You can then adapt them to suit you better and in ways that I could not have imagined. Most important though, I want you to have a real desire for looking toward future methods and thereby creating fresh and exciting work for the Web.

Why do you need Transcendent CSS?

I'm a designer. I like to design stuff. Some days I wish I designed iconic stuff such as classic cars or maybe the Apple iPod—stuff that people love and that makes me piles of cash…enough cash to buy as many classic scooters and 1960s Minis as I can fit in my garage. But you see, for one, I don't have a garage, and for another I enjoy what I design too much. Call me Mr. Obsessed if you like, but I just love designing for the Web.

I haven't always enjoyed the Web so much. Many times in my design career I could have cheerfully put down my computer after days of frustration and gone to do something completely different. Sometimes after struggling with one problem or another, the thought of spending hours in a garden shed with nothing but an old radio for company and growing gigantic leeks seemed appealing indeed. But rather than talking to vegetables, I stuck to talking to myself, and before too long, it was "problem solved." My passion for the Web was back. Funny, though—I never expected Web design would be so challenging; I mean, it's not like climbing Mount Everest. People don't choose to do it just "because it's there."

But many parts of the Web design process can be challenging for designers like me who are visual thinkers. Every day we use visual tools such as Adobe Photoshop, Macromedia



Fireworks, and others to move pixels around a screen to achieve our design goals. Some of the more technical aspects of the stuff that makes Web sites work today, particularly writing meaningful markup and CSS, can be unfamiliar or even seem counterintuitive.

CSS is not designer friendly

One factor is that as a technology built to help solve design problems, CSS is not very designer friendly largely because it was created for designers by technologists rather than by other designers.

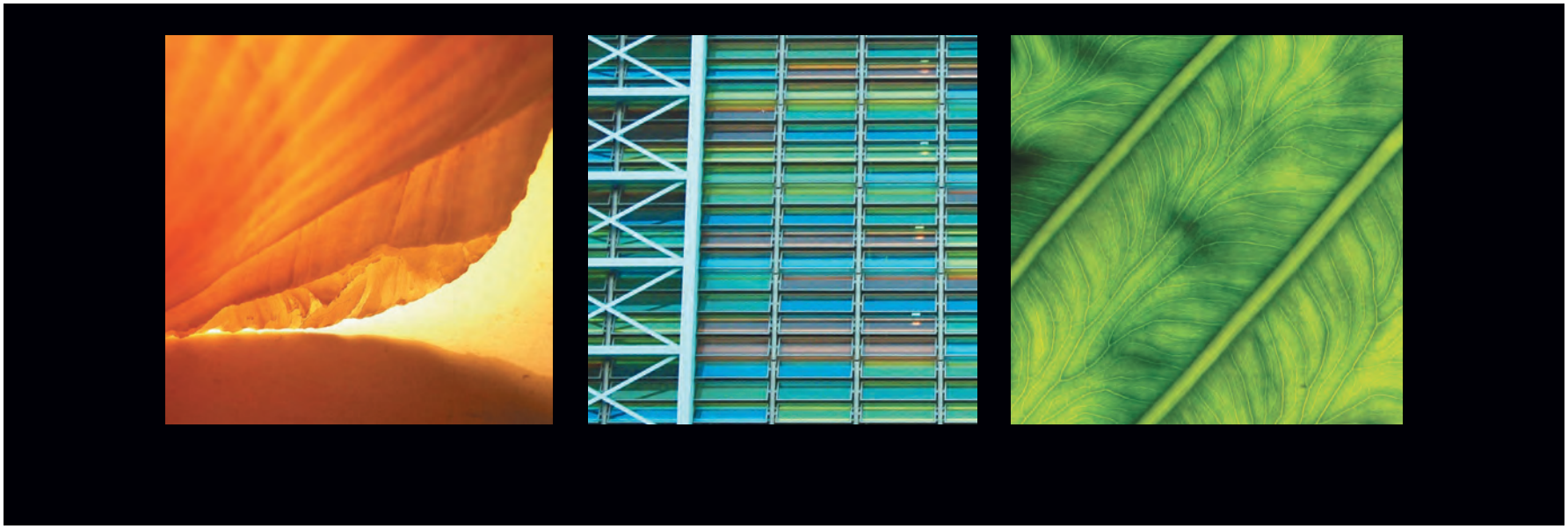
Although the basic principles are simple enough, as you can see here:

```
p { color : #000; }
body { background-color : #fff; }
```

for some designers, terms such as *the cascade* and *inheritance* are more difficult to understand. Add to this discussion talk of *positioning*, *collapsing margins*, or the *box model*, and you might see one of the many reasons why designers have taken a reasonably long time to adopt CSS.

For the longest period of time after CSS was launched, it was very much the domain of technologists. Their big brains were better equipped to understand concepts such as *specificity* as well as the myriad of largely unintelligible CSS hacks that were necessary to implement a design more or less consistently across different browsers.

These difficulties have done little to reduce the knowledge gaps that have always existed between visual designers and the technical developers who work to implement designs using code, and they have often left designers feeling frustrated with CSS.



Why visual Web editors are failing designers

Although over the years visual Web editors such as Macromedia Dreamweaver have improved the standard of the markup and CSS they generate, none of them has adequately solved one of the major problems facing visual designers who work on the Web, that of relating what they see in their designs to the meaningful markup that will make it possible.

It is now an essential part of a professional Web designer's job to understand the fundamentals of meaningful markup and CSS. Visual Web editors, including Dreamweaver, need to help designers see "beneath" their designs and "visualize" their code.

Web browsers' rendering inabilities have stifled progress

And then of course Web browsers challenge us. From the earliest implementations of CSS in Microsoft Internet Explorer 3 (the first mainstream browser to support any CSS), working with CSS has often been exasperating. Browser bugs, rendering errors, or just the plain stupidity of certain browser behaviors all made our lives more difficult.

This situation did get better over time; Netscape 4.*x* was better than Internet Explorer 3, and Internet Explorer 5 for the Mac showed for the first time that CSS could "work." Internet Explorer 6 had the best CSS support of any browser when it was released, despite its now well-known catalog of colorfully named bugs.

I imagine my fourteen-year-old son, now taking his first steps into Web design, might look back in years to come and laugh when he reads about the "double-margin float" or the "peekaboo bug." He also might wonder whether the "3px jog" was how designers today exercised before they ate breakfast.

Never underestimate the power of the individual

Still, a way was found; largely because of the dedication of developers including Tantek Çelik, Todd Fahrner, and Eric Meyer whose work then made it possible for designers such as Douglas Bowman and Dave Shea to show that working with CSS was not only desirable but a practical reality. Throughout the years since the first CSS specification was released, dedicated people like this all over the world have battled with and found workarounds for almost all the problems designers working with CSS face on a daily basis.

It is also important not to forget that by working on liaisons with browser vendors and software developers and also working in education, members of the grassroots Web Standards Project—including Rachel Andrew, Molly E. Holzschlag, and Dori Smith—have all played major roles in raising awareness of the importance of standards.

Without these individuals working separately in small groups, CSS use would never have been the powerful Web design tool it is today.

Relatively speaking, today we have it easy when compared to the pioneering early days of CSS. Many new Web designers will never have experienced working with table-based layouts or the frustrations of getting CSS layouts to work in what we think of now as ancient browsers such as Netscape 4.

Some accessibility sites are downright ugly, but the problem lies with those sites' designers and not with accessibility, which carries no visual penalty. The same is true for Web standards, even if the look and feel of the W3C Web site is unlikely to motivate designers to get busy learning about XML or CSS2.

—JEFFREY ZELDMAN
Designing With Web Standards. First Edition, May 2003

Expanding the creative possibilities

Now that Web browsers have reached a certain level of maturity in their support for standards such as CSS, they provide us with a firmer foundation on which to develop our designs. It is time to move forward, not stand still.

The mechanics of table-based layouts confined our designs to a rigid grid and reinforced time after time the conventions of the typical two- or three-column layout still seen on countless Web sites. CSS offers new creative possibilities by using floats and different forms of positioning, it offers layering in the form of the z-index, and it gives you the power to style any element through the CSS box model. These opportunities for creativity were not possible with table-based layouts (**Figure 1.1**).

Designs that adapt beyond the screen

Two years before the publication of the first CSS specification and a full six months before Netscape launched its first Web browser, Chris Lilley, who would become the chairman of the W3C (World Wide Web Consortium) CSS Working Group, essentially predicted what would happen to the Web during the rest of that decade:

> *If style sheets or similar information are not added to HTML, the inevitable price will be documents that only look good on a particular browser, at a particular window size, with the default fonts, etc.*
>
> —Chris Lilley (former chairman of the CSS Working Group, at the time snappily called the Style and Formatting Properties Working Group), May 1994

After many years of hard work, the Web has now finally made its way onto mobile phones, gaming devices, and televisions; in the future, this will include all manner of other portable devices that haven't been invented yet.

Hold that download for a moment; I'm going into a tunnel

The truth of the matter is that we simply do not know where the Web will crop up next. In years to come, my son will laugh at the idea that the Web was ever "hardwired" to the desktop, just as I laugh at the memory of my first mobile phone that came with a battery pack the size of a house brick and weighed just about the same.

1.1 **Transitioning to CSS-based layouts does not always mean better structured or ordered code. Left: The presentational content order of a table-based layout. Right: The tables have been replaced by `<div>`s without reordering the content.**

In such a rapidly changing medium, Lilley's words now ring truer than perhaps they did when he first wrote them. Today's designers should at least be aware that their designs will have to adapt to the many needs of these different environments. Lightweight, meaningful documents and CSS are key factors in successfully transitioning a design from the desktop to other devices, be those devices printers, small-screen handheld computers, or personal media players or mobile phones (**Figure 1.2**).

Accessibility is design, not a feature

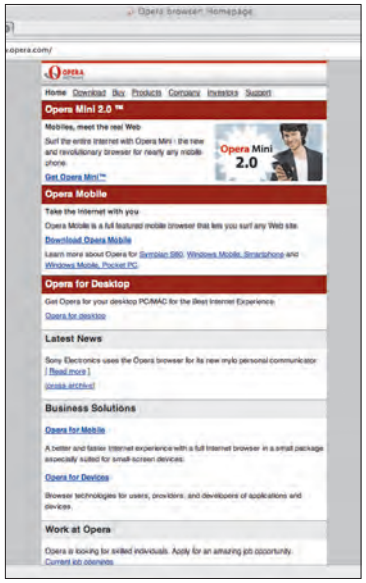
Good information architecture, usability, and accessibility have rightly become areas of concern for professional designers and developers. In particular, ensuring the widest possible accessibility is not only an ethical issue but also a commercial one. However, many designers, developers, and other specialists in the accessibility realm have wrongly limited conversations about accessibility to merely serving the needs of people with disabilities.

Accessibility is a matter of usability

Much of what has been written about accessibility has focused on ensuring that sites are accessible to people who are blind or visibly impaired. Much more has been written about ensuring that sites comply with accessibility guidelines or in some cases laws. However, most of this simply misses the point.

Although serving the needs of people with disabilities should of course be a concern, the far wider issue—that accessibility is a matter of usability—has rarely been discussed. As designer professionals, we should be designing our content so it is globally accessible and meets the needs of as many people as is possible and practical given our specific circumstances, regardless of their abilities or the type of device they choose to access the Web.

For the traveling businessman, whether he can successfully log on to his company's intranet to check sales figures on a handheld computer is both a usability and an accessibility issue, as is that many movie sites offering branded goodies for your mobile phone do not offer you the ability to access those pages using a mobile phone. It's important to realize that it is through good design that you can remove as many barriers to access for as many people as possible.



Opera Mini™ simulator

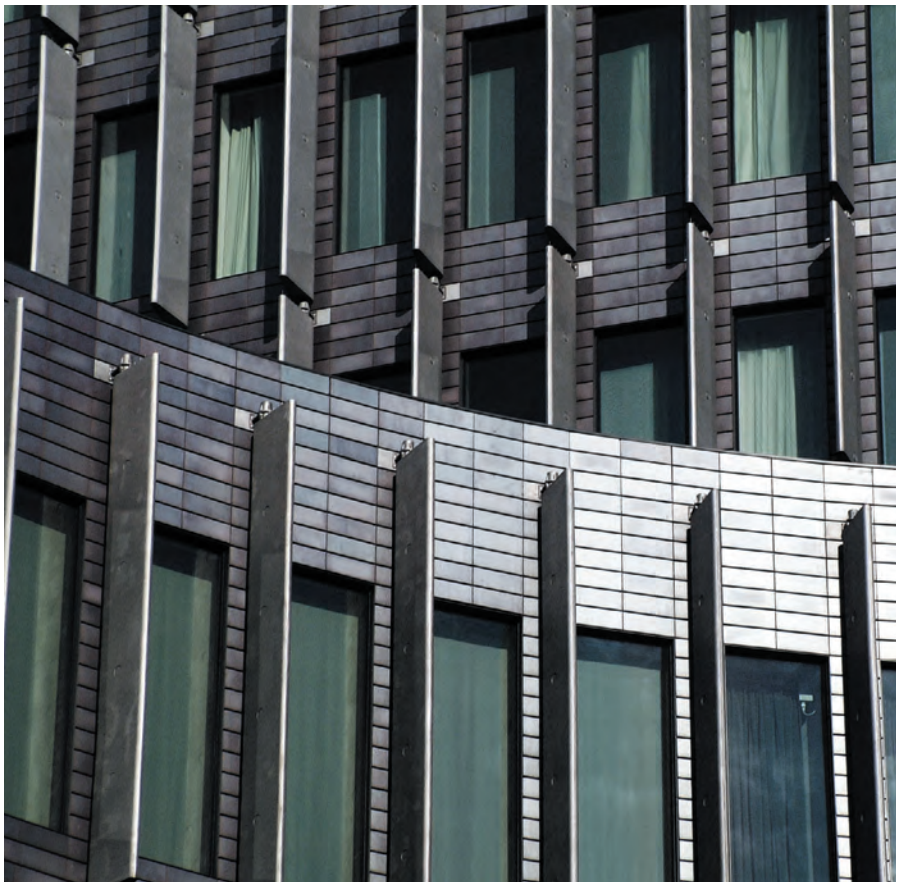
1.2 **Using CSS everywhere**

Wearing badges is not enough in days like these

Unfortunately, many designers still view accessibility concerns as limitations on their creativity, guidelines they should comply with, or laws they should obey rather than as an everyday part of the job of a designer. Accessibility has too often been viewed as an external factor with sites tested after completion to ensure that they meet one standard or another:

> *If you look at accessibility as a feature, you'll probably leave it out. Most developers are gonna leave it out anyway; most developers don't know the first thing about accessibility or even that it's important.*
>
> —Joe Clark (http://joeclark.org/ice/iceweb2006-notes.html)

It is sad that much of the work involved in ensuring that content and services are accessible at best currently takes place late in the design and development processes and at worst is an afterthought that requires a refit. When given the choice between the latest exciting Ajax interface and accessibility, many companies will choose the dynamic interface and may plan accessibility testing or features as part of a future release.



Explaining Transcendent CSS to your clients

Many designers have been itching to use the full power of CSS in their work but have been held back because they believe their clients will expect their designs will appear the same across all browsers.

It is true that many clients, companies, and hiring organizations do expect it is part of a designer's job to ensure such cross-browser compatibility. Although it rarely should be a designer's job to educate a client, some simple analogies can help you explain in broad terms the concepts of Transcendent CSS to your clients.

In many other areas of design and technology, the concept that the functionality or user experience of a product is the same regardless of the age or competence of a technology is ridiculous. Consumers not only expect that technologies will improve over time, but they also want to know they have bought the most up-to-date product.

You can easily explain to a client that you design for the most modern browsers using the latest coding techniques, but you still provide a good experience for people who use outdated browsers.

HIGH-DEFINITION TV

HDTV (high-definition television) offers a far-higher-resolution picture and better-quality sound than conventional television for people using HDTVs and receivers. High-definition broadcast television has been available in the United States for far longer than it has been in the United Kingdom. Since the first satellite broadcasters announced they were beginning high-definition broadcasts in the United Kingdom in 2006, electrical retailers and makers of televisions have been clambering to jump aboard the "HD-ready" bandwagon.

If you choose not to upgrade your equipment to HDTV or not to pay an additional subscription to view in high-definition, you are not excluded from watching your favorite soap, cop show, or football game; you simply see a slightly inferior but perfectly acceptable picture quality because your hardware is less up-to-date.

THE IPOD FACTOR

When Apple first launched its iPod portable music player, the player had fewer features and a different user experience than buyers of more recent models now enjoy. In a highly competitive marketplace, Apple has maintained its lead by providing new features with almost every release, such as the capability to store album art and play music videos or TV shows downloaded from its iTunes Store.

Older versions of the iPod still provide the same functionality they did when they were first unboxed. Consumers would never expect that an older player would offer the same facilities as a newer model, and they accept that to gain access to the new features they must upgrade.

PERPETUALLY UPGRADING YOUR SOFTWARE

The software industry has typically been littered with buggy, unstable software releases and operating systems that would freeze or crash on an almost daily basis. Yet despite these failings, consumers accept that a newer, better version will be just around the corner and are prepared to upgrade.

As well as being people who commission designers to develop Web sites, clients are also consumers who are exposed to changing technologies almost daily, and as such they can easily understand that people using more modern browsers will get an enhanced user experience or design.

Note

In June 2005, I expanded on my opinion that in order for the needs of people with disabilities to be better served on the Web, government-sponsored laws and regulations were counterproductive. "Accessibility and a society of control" sparked many interesting comments about how best to move the cause of Web accessibility forward (www.stuffandnonsense.co.uk/archives/accessibility_and_a_society_of_control.html).

However, when you work from the content out, using well-ordered and meaningful markup to provide the content and structure and using CSS for visual styling, you can build accessibility directly into the design process from the start. This will benefit everyone, including the designer who will be less challenged by accessibility issues, the client who will ultimately save money on retrofitting and have happier customers, and, most important, the visitors to those sites who will more easily get the stuff done that they need to get done.

Moving toward Transcendent CSS

Compared to the freedom enjoyed by print and multimedia designers, those of us who choose to work with markup and CSS have always suffered from factors limiting our designs; these factors are often beyond our control.

Since the first CSS specification was published, our creative potential has been hampered by the limited performance of browsers—unless, that is, we are prepared to sacrifice the purity of our documents by using presentational markup or choose to work solely in Flash.

Browser makers have mostly continued to improve their support for Web standards, particularly CSS and the W3C DOM (Document Object Model), but Microsoft's prior decision to allow development of its Internet Explorer browser to stagnate at version 6, plus the widespread, though thankfully now, reduced usage of its older versions has led to designers being hesitant about using some of the more advanced aspects of CSS.

When competing browsers such as Mozilla Firefox, Mozilla Camino, Apple Safari, and Opera began to shave percentage points off Internet Explorer's market share (**Table 1.1**), forward-thinking designers began to investigate ways to reward users of these more modern browsers by giving them an extra layer of design finesse. This technique became known as *progressive* or *MOSe enhancement*.

Table 1.1 Browser usage in July 2004–2006

	Internet Explorer 7 (beta)	Internet Explorer 6	Internet Explorer 5	Firefox	Mozilla	Netscape	Opera
2006	1.9%	57.8%	4.2%	25.0%	2.2%	0.4%	1.4%
2005	–	67.9%	5.9%	19.8%	2.6%	0.5%	1.2%
2004	–	67.2%	13.2%	–	12.6%	1.8%	2.0%

Source: W3 Schools (www.w3schools.com/browsers/browsers_stats.asp), August 2006

MOSe enhancements

Way back in 2003, Dave Shea, the Canadian designer, author, and creator of the CSS Zen Garden, wrote about a compelling new approach to the problem of creating designs for the differing capabilities of competing Web browsers; he coined this approach MOSe (Mozilla, Opera, Safari enhancement).

With MOSe, Shea claimed it was acceptable for designers to exploit Internet Explorer 6's (and earlier edition's) lack of support for certain CSS selectors (among them child, adjacent sibling, and attribute selectors) to provide an enhanced design for users of modern browsers that was layered on top of a standard design that was visible to all.

In theory, Shea's suggestion was no more complicated than the `@import` at-rule technique. This widely used technique exploited that Netscape 4 could not interpret `@import` and prevented the aging browser from CSS that it either could not understand or bungled, but this technique has one important difference. When Shea wrote his article, Internet Explorer 6 was, despite its many flaws, a browser capable of supporting most of the CSS specification. It was the most widely used browser in circulation, so any attempt to design away from it had to be a subtle one.

Working around Netscape 4.*x*

Netscape 4.*x* had some CSS support. In an attempt to shield CSS styles that Netscape 4.*x* could not interpret, many designers chose to hide all but the most basic styles by using the `@import` at-rule.

This is a basic style sheet visible to all browsers including Netscape 4.*x*:

```
<link rel="stylesheet"
type="text/css" href="simple.
css" />
```

This is a more advanced style sheet for browsers with more explicit CSS support:

```
@import "modern.css";
```

All browsers will load simple.css; however, only browsers that understand the `@import` at-rule will load modern.css. Since modern.css is imported after simple.css, its rules will override those in simple.css unless those rules are more specific.

The key to the MOSe method is somewhat similar to how NN4 (Netscape Navigator 4.x) page design developed as CSS became more prevalent. After creating a basic, functioning page in IE, you add extra functionality with these selectors.

—DAVE SHEA
www.mezzoblue.com/archives/2003/06/25/mose/

FROM THE CSS ZEN GARDEN

Building on the MOSe approach, This Is Cereal, Shaun Inman's CSS Zen Garden design, used the CSS selectors that weren't supported in Internet Explorer to provide visitors using modern browsers with a richer experience. Inman transformed unordered lists of links into subtle drop-down menus with alpha-transparent PNG images. Although he was not the first designer to use this technique, his results were inspirational.

Visitors using older browsers see styled but plainer unordered lists and are not aware that an alternative version exists. Subtle MOSe enhancements such as this have since become a common feature in many of CSS Zen Garden's designs (**Figure 1.3**).

1.3 Featuring MOSe enhancements on CSS Zen Garden

AND ALL THAT MALARKEY

Inspired by Inman, for my personal Web site I created two distinct designs: one full-color design inspired by the British mod(ernist) music movement of the 1960s and one black-and-white 20ld design that was inspired by the stark two-tone imagery made famous by the British ska record label 2Tone.

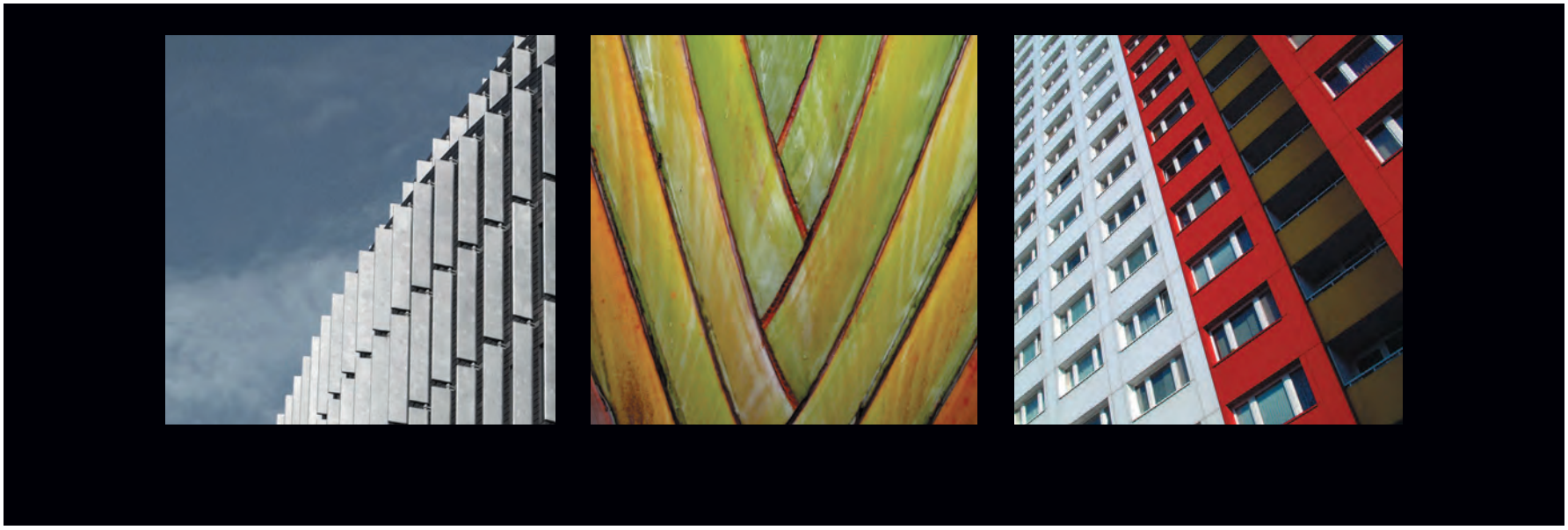
Because ska music came before mod, I made the ska design available only to legacy versions of Internet Explorer. I implemented the mod design using CSS2.1 selectors that are understood only by mod(ern) browsers, which was a fun way to highlight the different capabilities of browsers.

Note: My decision to "punish" visitors using Internet Explorer by depriving them of the full mod design was not altogether well received, as you can see for yourself at www.stuffandnonsense.co.uk/archives/and_all_that_design_malarkey.html.

Progressive Enhancement

Several months after Shea's MOSe article appeared, another was published in *Triangle TechJournal* that further explained the concept of progressive enhancement:

> *Progressive Enhancement presents a viable approach by enabling the delivery of information as demanded by the users, while embracing accessibility, future compatibility, and determining user experience based on the capabilities of new devices.*
>
> —Debra Chandra and Steve Champeon (http://hesketh.com/publications/progressive_enhancement_paving_way_for_future.html)



But fully adopting the methods of progressive enhancement has been difficult for designers to achieve in commercial projects until now. This difficulty stemmed not only from the commanding market share of Internet Explorer 6 but also from the belief that it is correct to set a design benchmark based on the most popular browser, even if that browser is less capable and less advanced in its support for modern standards than its competitors.

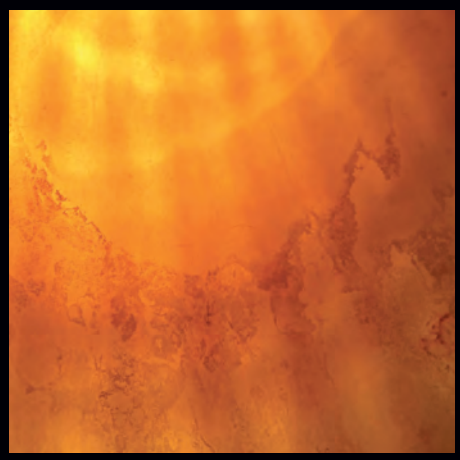
For such a young and dynamic medium as the Web, the notion that designers should not push design boundaries forward because of only one browser, even when that browser is the market leader, seems incompatible with progress.

Both MOSe and progressive enhancement were ideas intended to encourage designers to use all the tools made available to them in the CSS2.1 specification for browsers that supported these selectors and rules. Even today, several years later, these CSS rules are often described as *advanced* or *cutting edge* despite that they were invented only a few years after the birth of the commercial Web.

MOSe and progressive enhancement have been used on personal portfolios and blogs; you can rarely find them on mainstream, commercial projects. As a result, progressive Web design has largely stalled and, if left much longer, will begin to go stale.

My question is, can progressive enhancement still be called *progressive* three years after the term was first coined? The answer must be no, so it is time to move forward.

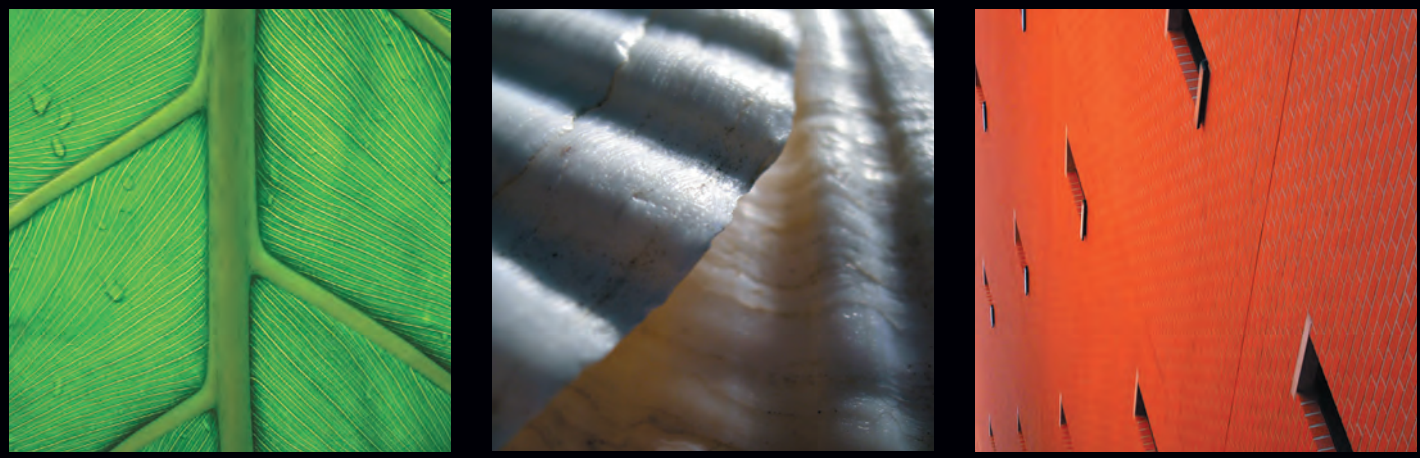


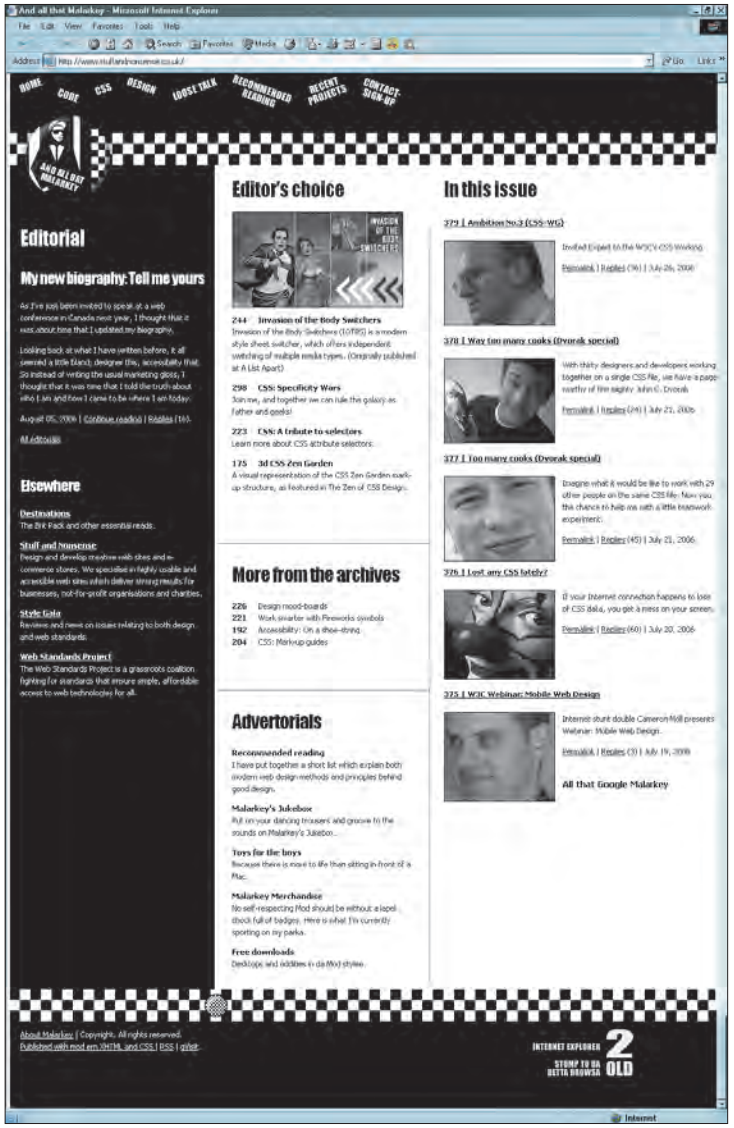


The Principles of Transcendent CSS

This brings us to the subject of this book. The principles of Transcendent CSS allow Web designers to focus on their creative goals without being preoccupied with technical constraints. These principles allow Web designers to look to the future without being compromised by the limitations of the past.

1. **Not all browsers see the same design.**
2. **Use all available CSS selectors.**
3. **Use CSS3 where possible to look to the future.**
4. **Use JavaScript and the DOM to plug the holes in CSS.**
5. **Avoid using hacks and filters.**
6. **Use semantic naming conventions and microformats.**
7. **Share your ideas, and collaborate with others.**





1.4 Sending different designs to different browsers at All That Malarkey

1 Not all browsers see the same design

Whereas progressive enhancement begins with less capable browsers such as Internet Explorer 6 and then uses CSS selectors to add functionality, Transcendent CSS abandons the notion that a less-capable browser is the benchmark.

Transcendent CSS reverses the Progressive Enhancement approach that creates a design that can be rendered by all browsers but is limited to the capabilities of the lowest common denominator.

Transcendent CSS sets that benchmark squarely where it belongs today, with the CSS2.1 specification and those browsers that support it. It uses all the available CSS2.1 features, not to add visual enhancements but to accomplish the best design for the most, standards-capable browsers (**Figure 1.4**).

In practice, this approach will result in some visitors seeing a reduced design—how much of a reduced design of course depends on your preferences and the specific needs of the audiences using the sites you create.

2 Use all available CSS selectors

Transcendent CSS uses all CSS2.1 selectors plus other CSS features including pseudo-elements and dynamic pseudo-classes. These selector types include the following:

Attribute selectors

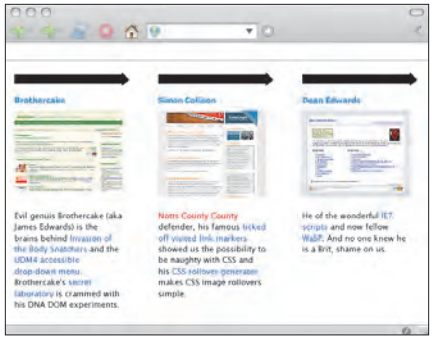
Attribute selectors are amazingly powerful; they offer ways to style an element either based on whether an element has an attribute name such as `href` or based on the attribute value such as `"http://www.stuffandnonsense.co.uk"`.

In the following examples, all images that contain an `alt` attribute will have a gray border (**Figure 1.5**):

```
img[alt]  {
border : 1px dotted #999;
}
```

```
<img src="http://www.hicksdesign.co.uk/images/love/bp.gif"
alt="Brit Pack: A proud member" />
```

1.5 Outlining images with an `alt` attribute



1.6 Highlighting anchor with `title` attributes

1.7 Styling a class `rant`

1.8 Floating a main content division

And all anchors that contain a `title` attribute will be red (**Figure 1.6**):

```
a[title]  {
color : #c00;
}

<a href="http://www.collylogic.com/" title="Simon Collison">
Former Notts County Defender
</a>
```

You can also apply styles to an element based on the content of its attributes (**Figure 1.7**):

```
p[class="rant"]  {
font-weight : bold;
text-transform : uppercase;
}

<p class="rant">
But I am done apologizing. I am realizing more and more that the real world
produces shocking HTML and CSS is inherently very difficult to organize at the
best of times, let alone to organize well within a project that is jumping all
over the place at an extraordinary pace.
</p>
```

(**Figure 1.8**)

```
div[id="content_main"]  {
float : left;
}

<div id="content_main">
<blockquote>
<p>I hope that things will change. I hope that some young guns will take up
the challenge, stop following the crowd, and really push CSS to its fullest
potential.</p>
<p>Jeremy Keith</p>
<blockquote>
</div >
```

With pattern matching, you can style an element based on only part of its attribute, in this case the base URL of the quotation's citation (**Figure 1.9**):

```
q[cite*="http://www.andybudd.com/"]  {
padding-left : 100px;
background : url(images/budd.jpg) no-repeat left top;
}
```

```
<q cite=" http://www.andybudd.com/archives/2006/07/layout_grid_bookmarklet/">
Inspired by Khoi Vinh's post about using a background image of a grid for
layout, I decided to knock up a quick Photoshop style Layout Grid Bookmarklet
</q>
```

You will learn much more about attribute selectors and their practical applications in Part 4, "Transcendence."

1.9 **Adding a background image to a quotation**

Child selectors

A *child selector* targets a direct child of a given element. For example, this gives you the potential to style anchors that are direct children of list items differently from other anchors on a page. Child selectors consist of two or more selectors separated by the `>` combinator.

> **Note:** *Combinators* separate two or more selectors that make up a *combined* selector. Available combinators include white space, `>`, and `+` as well as a comma or a colon.

This rule will style all anchors that are children of a `<div>` element:

```
div > a {
text-indent : -9999px;
}
```

This rule affects only `<a>` elements that are direct children (not other descendants) of `<div>` elements. If any other elements appear between the `<div>` and the anchor—for example, a `<ul>` element—the selector will not match, and the `text-indent` style will not be applied (**Figure 1.10**):

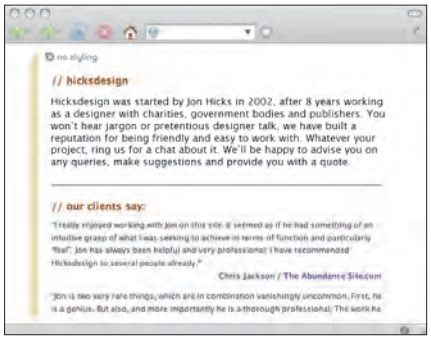
```
<div>
<a href="#content">Skip to content</a>
<ul>
<li><a href="http://www.andybudd.com">Andy Budd</a>
<li><a href="http://www.adactio.com">Jeremy Keith</a>
<li><a href="http://www.clagnut.com">Richard Rutter</a>
</ul>
</div>
```

1.10 **Hiding a "skip to content" link**

Adjacent sibling selectors

An adjacent sibling selector consists of selectors separated by the + combinator. It matches an element that is the next sibling to the first element. Elements must have the same parent, and the first must immediately precede the second:

```
h2 + p {
font-size : 110%
border-bottom : 1px solid #666;
}
```



1.11 Sibling selectors

When applied in the following example, the previous rule will affect only the first paragraph (**Figure 1.11**):

```
<h2>Hicksdesign</h2>
<p>Hicksdesign was started by Jon Hicks in 2002, after 8 years working
as a designer with charities, government bodies and publishers.</p>
<p>You won't hear jargon or pretentious designer talk, we have built
a reputation for being friendly and easy to work with.</p>
```

Pseudo-classes and pseudo-elements

You can use pseudo-classes and pseudo-elements to style elements based on information that is not available in the DOM. For example, it is often desirable to style the first line of a paragraph or the first letter of a heading.

PSEUDO-CLASSES

Pseudo-class style elements are based on characteristics other than their identifier, attributes, or content.

1.12 First-child pseudo-class

This **:first-child** pseudo-class matches an element that is the first *child* of another element. Imagine you want to give the first paragraph of a news article more visual prominence. If the article appears in a element with a class name of news, the following rule styles the first <p> element in each article (**Figure 1.12**):

```
div.news p:first-child {
font-size : 110%;
font-weight : bold;
}
```

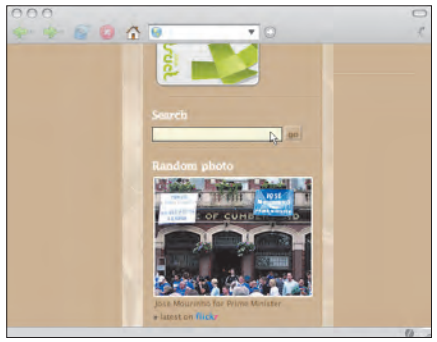
```
<div class="news">
<p>We are really pleased that this years d.Construct sold out in under 36hrs.
If you were lucky enough to secure a ticket, we look forward to seeing you in
sunny Brighton.</p>

<p>Brighton is a popular little spot and hotels get booked up early.</p>
</div>
```

A subset of pseudo-classes is the *dynamic* pseudo-class. These are pseudo-classes that have some dynamic feature. You can use dynamic pseudo-classes to style elements depending on certain actions that a site visitor might perform.

`:focus` applies while an element has the *focus*, such as when a visitor either clicks within a form or tabs to a form input (**Figure 1.13**):

```
input[type=text]:focus {
color : #000;
background-color : #ffc;
}
```



1.13 Focusing on an input

You can use the **`:lang`** language pseudo-class to style elements where content is in a particular language, perhaps a language that differs from the main language of the document. For example, the following rule applies a small German flag icon to any `<blockquote>` in the German language (**Figure 1.14**):

```
blockquote:lang(de) {
padding-right : 30px;
background: url(images/de.png) no-repeat right top;
}

<blockquote lang="de">
<p>Die Webkrauts setzen sich dafür ein, die Vorteile der Webstandards auch
im deutschsprachigen Raum stärker zur Geltung zu bringen. Wir leisten
Aufklärungsarbeit durch Veröffentlichungen im Netz und in anderen Medien.</p>
</blockquote>
```

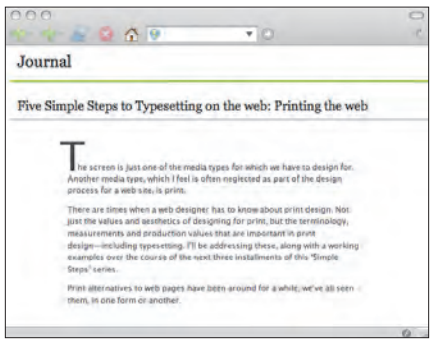
1.14 Adding a language flag

PSEUDO-ELEMENTS

Pseudo-elements also allow you to style parts of the document that are not available as nodes in the DOM.

The **`:first-line`** pseudo-element targets the first line of a paragraph of text. The number of words in any first line will vary according to the scaling of the text size in the browser.

1.15 **Styling the first line**



1.16 **Introducing the first letter**

The following rule applies to the first line of text in any paragraph on a page (**Figure 1.15**):

```
p:first-line {
font-size : 120%;
font-style : italic;
}
```

The **`:first-letter`** pseudo-element allows you to target the first letter or digit of an element. The next rule applies to the first character in a `<p>` element with a class name of `introduction` (**Figure 1.16**):

```
p.introduction:first-letter {
font-size :400%;
font-weight : bold;
}
```

3 *Use CSS3 where possible to look to the future*

Although CSS2.1 is, at the time of this writing, still not yet a final recommendation and the W3C's CSS Working Group has much to do before completing its work on all aspects of the CSS3 draft modules, evolving browsers such as Firefox already support some of these new technologies. When practical, Transcendent CSS makes it possible to use some of the already implemented CSS3 features. These include parts of the CSS3 multicolumn module that is already supported by Firefox or multiple background images that are currently enabled in Safari.

Although it is too early to rely heavily on CSS3 features in everyday design, it is important to use what is practical today as a way of understanding the creative opportunities of what will be possible tomorrow. You will see many of the exciting possibilities of CSS3 in Part 4, "Transcendence."

4 *Use JavaScript and the DOM to plug the holes in CSS*

One of the ways designers and developers can work around the limitations of browsers is by using JavaScript and the DOM to plug some of the holes in CSS support. This technique has been made popular by JavaScript developers including Cameron Adams and Dean Edwards (see Part 4).

5 *Avoid using CSS hacks and filters*

Hacking and filtering CSS has been a necessary evil since the earliest days of CSS layouts. From the first media HTML filter that was used to hide CSS from Netscape 4.*x* to the (in)famous box model hack, using hacks and filters has become an almost everyday necessity with which to handle the inconsistencies in some browsers' treatment of CSS.

Just like your markup, hacks should be valid

In 2002, Tantek Çelik's box model hack, in all its ugly glory, made CSS layouts consistent across all browsers. It did this by working around the fact that, at the time of its invention, Internet Explorer for Windows still incorrectly calculated the width and height of a box:

```
div#content {
width : 400px;
voice-family : "\"}\"";
voice-family :inherit;
width : 300px;
}
```

These unfamiliar `voice-family` properties hid the second, correct width of a content area from Internet Explorer for Windows by giving it an incorrect width and then confusing it with valid declarations that it could not interpret. Browsers that supported the `voice-family` property then implemented the correct CSS width. CSS layouts were, as a result, finally practical.

Despite its unfamiliar syntax, the box model hack contains valid CSS, one of the key principles in a transcendent approach to using CSS hacks or filters.

Toward the end of 2005, Çelik wrote a seminal article, "Pandora's Box (Model) of CSS Hacks and Other Good Intentions," that recapped the history of CSS hacking and recommended best practices. Çelik's article forms the basis of the Transcendent CSS approach to using CSS hacks—that you should avoid using them at all except as a last resort.

If hack use is unavoidable in any given situation, hacks should target either only those browsers that have been abandoned, such as Internet Explorer for the Mac, or browsers frozen in their development but still in wider circulation. This now includes Internet Explorer 6 for Windows. You should always avoid using hacks that target a current version of any browser.

Generated content using `:before` and `:after`

The benefit of generated content is often debated between designers and developers who believe CSS should only style content on a page and not add content to it and those who believe it has a rightful place in CSS.

You can use the **`:before`** and **`:after`** pseudo-elements to insert generated content either before or after an element's content. For example, you can display the `href` attribute of a link using this:

```
a:link:after {
content : " (" attr(href) ") ";
}
```

The next rule applies to every second-level heading and inserts a decorative image before the content:

```
h2:before {
content : "";
display : block;
height : 20px;
width : 20px;
background : url(target.png)
no-repeat 0 0;
margin-right : 20px;
}
```

CSS-generated content is supported by standards-aware browsers but will not be supported by Internet Explorer 7 or perhaps even future versions of Internet Explorer.

Two popular DOM scripting plugs

Two very popular DOM scripting plugs are Cameron Adams's resolution-dependent layouts and Shaun Inman's position clearing.

Resolution-dependent layouts: In early 2006, Adams published an experimental technique that uses JavaScript to detect the width of a browser window and load different CSS rules according to that width. This made it possible for designers to provide a slightly modified page layout for visitors using lower screen resolutions than for visitors using higher resolutions such as 1024x800 pixels. You can find out more about the resolution-dependent layout technique at www.themaninblue.com/writing/perspective/2006/01/19/.

Inman position clearing: To solve the problem of footers not "clearing" absolutely positioned columns, Shaun Inman developed an ingenious JavaScript and CSS solution to force footers to drop below absolutely positioned content. You will be using Inman's solution in Part 2, "Process," and you can find out more about Inman position clearing at www.shauninman.com/plete/2006/05/clearance-position-inline-absolute.

Note: Tantek Çelik and Molly E. Holzschlag, among others, have written two excellent articles on how to manage hacks and filters. Çelik's "Pandora's Box (Model) of CSS Hacks and Other Good Intentions" provides a fascinating history of CSS hacks at http://tantek.com/log/2005/11.html, and Holzschlag offers sound advice in "Strategies for Long-Term CSS Hack Management" at www.informit.com/articles/article.asp?p=170511&rl=1.

The demise of CSS hacks and broken pages

In late 2005 with the beta version of the long-awaited Internet Explorer 7 browser released, Internet Explorer's Program Manager for Layout and CSS, Markus Mielke, asked designers and developers to abandon their use of CSS hacks altogether and switch instead to using Microsoft's proprietary conditional comments:

> *We ask that you please update your pages to not use these CSS hacks. If you want to target IE or bypass IE, you can use conditional comments.*
>
> —Markus Mielke (http://blogs.msdn.com/ie/archive/2005/10/12/480242.aspx)

Supported only by Internet Explorer for Windows, conditional comments make it simple to either target or bypass different versions of Internet Explorer by placing comments in the <head> portion of your XHTML document.

For example, to provide a common set of rules to all browsers but only a specific set of rules to all versions of Internet Explorer for Windows, you can use this:

```
<link rel="stylesheet" type="text/css" href="standards.css" />
<!--[if IE]>
<link rel="stylesheet" type="text/css" href="ie.css" />
<![endif]-->
```

Alternatively, you may need to target a specific version of Internet Explorer, in this example, version 5:

```
<!--[if IE 5]>
<link rel="stylesheet" type="text/css" href="ie5.css" />
<![endif]-->
```

With Internet Explorer 7, providing far better CSS support than version 6 or its older siblings, perhaps the most useful conditional comment targets only Internet Explorer version 6 and older:

```
<!--[if lte IE 6 ]>
<link rel="stylesheet" type="text/css" href="ielegacy.css" />
<![endif]-->
```

Since conditional comments can be used only in the markup layer, not from within CSS, and are proprietary only to Microsoft browsers, many designers prefer to avoid using them. They rely instead on hacks such as the `* html` hack that exploits a bug in earlier versions of Internet Explorer's CSS rendering.

With many such bugs and errors now corrected in Internet Explorer 7 and with conditional comments valid in XHTML, conditional comments are gaining more popularity. I now use conditional comments in all my projects.

6 *Use semantic naming conventions and microformats*

Unless you've been away for a while sewing mailbags at Her Majesty's pleasure, you will already know that the semantic naming of elements and attributes has been a hot topic for designers and developers.

Where once you might not have thought twice about labeling a paragraph with `class="big-black-text"`, it is now more widely accepted that presentational names such as `header`, `left`, or `red` that describe an element's look or position are poor choices.

Some examples of presentational versus nonpresentational naming include the following (**Figure 1.17**):

Presentational name	Meaningful equivalent
`#header`	`#branding`
`#sidebar`	`#content_sub`
`#footer`	`#site_info`

To date, little consistency exists in the names that designers have chosen for their attributes. That's not surprising because it is part of a designer's job to come up with cool new stuff. The idea of using the same names repeatedly isn't something designers like to do; in addition, the idea that they would use the same as other designers doesn't have much appeal either.

Note

You can find out more about Internet Explorer's conditional comments and how to target specific versions of that browser at http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/overview/ccomment_ovw.asp.

Note

The universal (*) selector targets all children of a given element; for example, `body * { padding : 0 20px; }`. However, because HTML is the *root* element in a document, it can have no *ancestors*.

Note

Former CSS Samurai John Allsopp has long advocated for the adoption of standardized element naming and has created WebPatterns, a site where you can contribute your own ideas on the subject of naming conventions (www.webpatterns.org).

Thinking of element names is one area of Web design where designers should try hard to fit in with their peers, though. Designs won't suffer from it, and it makes working within teams and even across companies a lot more intuitive.

Developers also appreciate naming conventions because they enable them to develop applications that can scrape content such as calendars or contact information from pages to create relationships between sites.

Developing naming conventions

It was again in 2005 that I first turned my thoughts to the subject of the names that designers were choosing to label their elements. Andy Budd wrote an article about this topic:

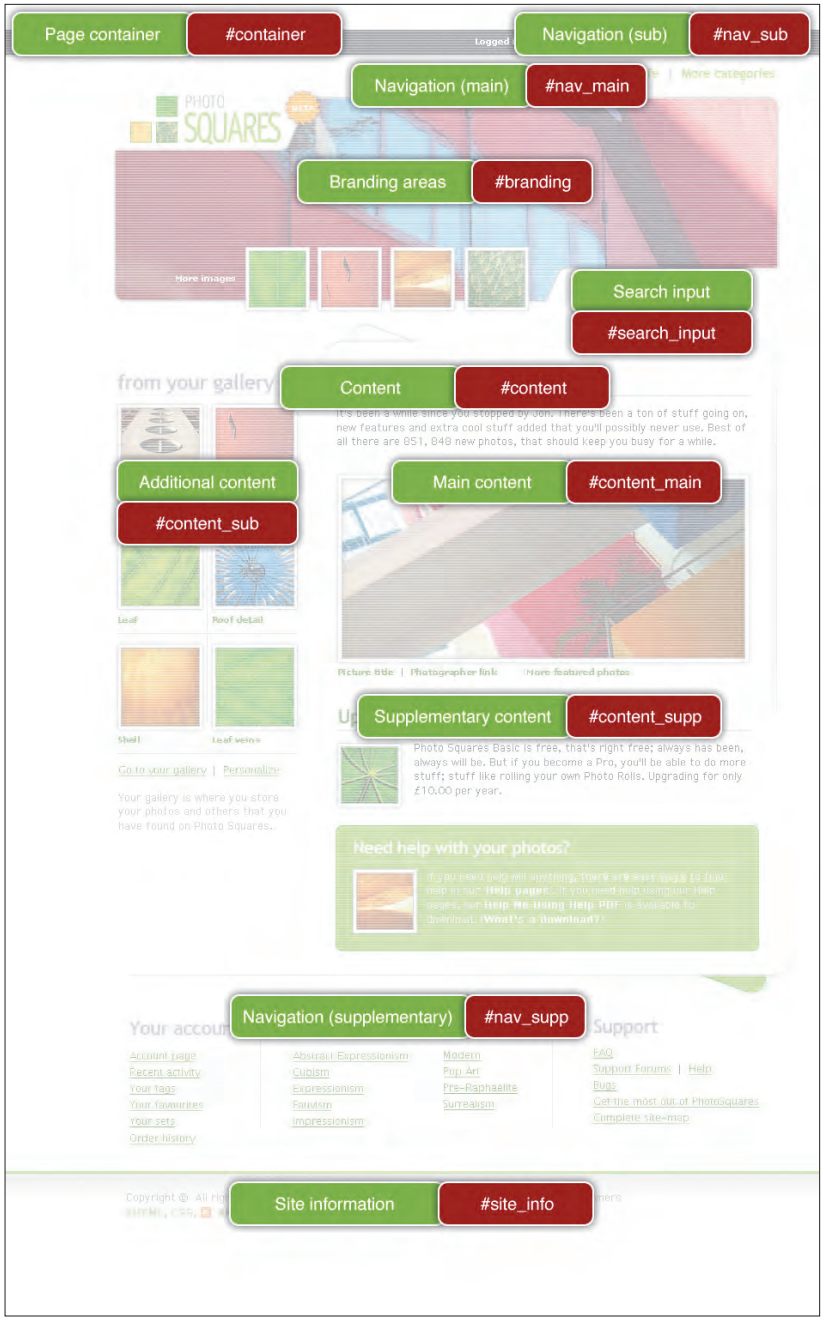
> *You'll wrap a div or a span around another element to act as a hook for your style. By doing this you're adding meaningless markup to your code for display reasons. This makes you feel bad, so you'll try to give these hooks some meaning. You'll put all your branding code inside a div called branding and your main content inside a div imaginatively called mainContent. The question that springs to my mind is, does it matter?*
>
> —Andy Budd (www.andybudd.com/archives/2004/05/semantic_coding)

Andy Budd's article and the readers' comments it inspired made me think about what the benefits of establishing conventions for element names might be. I soon realized such conventions do matter; in fact, they matter a lot.

I began by surveying the sites of forty designers and bloggers. I thought that if they worked similarly to the way I did, the names they had chosen for their personal sites would be reflected in their client work.

The result of these late nights of study revealed that at the time, attribute naming included wide variations, even for something as straightforward as the humble page container `<div>`. I found everything from the obvious `page` to `wrap` to Ethan Marcotte's wonderful `going-to-hell`.

When I wrote about this exercise on my Web site, it sparked several conversations elsewhere about why we should adopt common naming and whom it would ultimately benefit.



1.17 **Using semantic naming**

Note

You can read the results of my naming conventions survey at www.stuffandnonsense.co.uk/archives/naming_conventions_table.html.

Eric Meyer suggested in "Elemental Nomenclature," that should a convention be established, he would implement those attribute names on his own site in combination with his CSS *signatures* to give his visitors more control over his site layout and design.

Naming conventions help in teams

In team design and development environments, naming conventions can save your working and thinking time, because they leave you free to design rather than sit around wondering "Now what should I call this thing?"

Naming conventions can also help different people working on any one project to relate to each other, because they can more easily understand the many different elements and how those elements relate to their neighbors and to the wider Web.

When different people are working on a site at any one time, conventions make it simpler to understand the markup that has been written by another designer or developer. Conventional naming practices reduce the time it takes to mentally deconstruct a document and reduce the margins and costs of error.

Even if you work on your own rather than in a team environment, using naming conventions will help you, particularly when you return to a project after a period of time. I'm sure I'm not the only person who has opened a CSS file I wrote months earlier only to sit scratching my head while I wondered, "Umm `clear:left`; what does *that* relate to?" I'm sure you have done something similar.

In team-based work environments, you can extend your naming conventions from markup to other elements including CSS background images. In this instance, you can also use the name of an element for the filename of its background image. Rather than name an image file according to how it looks or what it contains, you can name it according to the element to which it relates. For example: `brighton_pier.jpg` might become `branding.jpg`, and `flowers.jpg` might become `body.jpg` because the new names relate directly to their context and not to their appearance.

Naming conventions can make for hours of geeky fun

Designers Douglas Bowman and Dave Shea made many geeks smile on April Fools' Day 2004 when they swapped their style sheets and stole each other's site designs. Although their swap was intended to be fun, Bowman and Shea were inadvertently making a serious point.

Does my site design not serve your needs, or bore you? Create something better suited to your tastes! I promise I won't mind; in fact, I'd like to see what you devise. If a set of ID naming conventions does firm up, I'll likely adopt it here so visitors can restyle my site consistently with others that use the same nomenclature. This is, it seems to me, the least I can do.

—ERIC MEYER, June 2004
http://meyerweb.com/eric/thoughts/2004/06/18/elemental-nomenclature

Note

You can read more about how they swapped designs at Dave Shea's Mezzoblue (www.mezzobluecom/archives/2004/04/02/poisson_davr) and Douglas Bowman's Stop Design (www.stopdesign.com/log/2004/04/02/return.html) articles.

Note

You can find the bastard offspring of the CSS Love Child at http://the-maninblue.com/experiment/CSSLoveChild.

Much of the effort that was involved in swapping designs was a direct result of the different `class` and `id` attribute names that each had chosen for their own site. Shea's page container `<div>` was labeled `container` while Bowman had chosen a `class` labeled `container`. The names for their side columns were also different, with Bowman's `sidecol` being incompatible with Shea's `sidebar`.

Had they chosen the same names, swapping their designs would have been far simpler. In the absence of common names, both were left with little choice but to edit the markup of each of their sites to make the swap possible (**Figure 1.18**).

SWAPPING STYLES WITH THE CSS LOVE CHILD

As well as being an entertaining example of the chaos that can ensue when you mix one person's markup with another's CSS, Cameron Adams's CSS Love Child demonstration highlights the tremendous benefits that could be achieved if designers chose the same element names as their peers.

TAKE YOUR VITAMINS

What if you love the content of *Vitamin* but prefer the typographical touches of *A List Apart* so you want to combine the two? By choosing the same element names, the designers of both these sites could allow a visitor to apply the *A List Apart* style sheet to *Vitamin*, putting them firmly in control (**Figure 1.19**).

1.18 **Swapping designs**

In an era where the social aspect of integrating content from many different sites is a key part of the Web 2.0 buzz, many more visitors would benefit enormously if designers and developers adhered to common naming.

Note: At the time of this writing, both Vitamin and A List Apart use attributes on their <body> elements, but neither gives visitors the control that either CSS signatures or common naming would provide.

A List Apart surprisingly uses an empty class placeholder, potentially a sleep-deprived oversight or an erroneous CMS artifact, while Vitamin limits its visitors by using a site-specific homepage body attribute. It would be far more helpful to visitors if both sites adopted both common naming and CSS signatures.

Common naming gives visitors extra control

Establishing these types of conventions has benefits not only for designers and developers but, most important, for visitors to the sites you create, giving them extra degrees of control over the sites they read on a regular basis.

What? Give visitors control over your pages? Yes, that's exactly what I mean, and it's sometimes a difficult concept for designers to grasp. After all, you're the designer, right? You know your Bézier curves from your CMYK conversions, so why should you let *them* tinker with your carefully crafted designs? The truth is that designing for the Web is unlike designing for other media. The Web is the first medium that gives people the option to change the way in which content is presented to them.

Do you enjoy the content of the *Times* but prefer the typography and layout of another newspaper? I'm sorry, but you have little choice but to stick with the *Times'* more traditional feel. The Web is a very different medium than newsprint, and designers must realize they are not designing sites for themselves but for the folks at home. You should make that experience as pleasant as you can by ensuring it is easy for your visitors to change something about your designs they may not like.

Sometimes a visitor might need only to increase a site's default text size. It would be far more convenient if Web designers made it easy for visitors to attach their own user style sheets to override particular styles by tapping in to a site's CSS signature. In reality, you rarely know what people are thinking about your sites, so you should provide them with the tools to display your content in any way they choose.

1.19 Letting users take control

Barbie's biggest fan sells her 4,000 dolls

By

FOUR thousand Barbie dolls which were collected by a fan over four decades are to be auctioned next month.

The dolls – most in pristine condition because they were never played with – were amassed by a fashion designer in Holland. They went on display at Christie's yesterday.

They range from Barbie No 1 (1959) – black pony tail, black and white swimsuit, pale and with heavy make-up – to Tango (2002), a box set of Barbie and "boyfriend" Ken dancing together. Ietje Raebel bought her first Barbie in the early 1960s for her daughter Marina but kept it for herself after realising the child would rather play with baby-like dolls.

By the time she finished collecting four decades later the doll had filled three rooms at her home.

Others among those who attend the London auction on Sept 26. The lots, priced from £70 to £1,200, are expected to fetch more than £100,000.

Mattel, the toy manufacturer, began selling Barbie dolls in 1959.

Mrs Raebel mostly bought them new and stored them in cabinets, wardrobes and the attic of her home.

Plantation Belle Barbie (1960s), Campus Sweetheart Barbie (1965), Fashion Luncheon Barbie (1966) and Sorbonne Barbie (1967) are among the lots being sold.

Twiggy Barbie (1967) is expected to attract fierce bidding, especially as the model has recently made a comeback with her poster and television campaign for [illegible]

COMMENT

Maybe there is something to global warming

SWITCH-OFF YOUTH FOR SIGNALS OF THE

M Night Shyamalan's new movie is the work of a filmmaker who has simply lost the plot

Absolutely flabulous

LAST NIGHT ON TELEVISION

Some people love taking on the world. In **Fat Beauty Contest** (C4) size-18 model Charlotte Coyle was determined to "prove to people that being curvy is equally as beautiful as being really slim". To which one could only say: go girl. To do so, she had "agreed" (with whom we were never told but a cynic might suspect [illegible]

for "plus-size" women. According to Charlotte such contests are all the rage in America so they should be here, too.

"People think beyond a certain size you're not allowed to be happy," said Charlotte. Certainly that seemed to be the case with many of the women who answered her call for contestants, some of whom mistook the audition for a tearful therapy session. (It *was* a tad insensitive to refer to the fifty or so women who turned up as a "stampede".) Many had suffered under the yoke of sizeism all their lives and were determined to cast it off. But even Charlotte wasn't immune to blind spots of [illegible]

"She's got a flat stomach," complained her chubby assistant Zoe, as if it were some kind of criminal offence. "You've hurt my feelings," responded Charlotte with exquisite model-logic. But wounded as she was, she had to relent – leaving the two contestants in the curious position of being barred for being too thin.

This was a show crammed with contradictions anyway. If you think it is wrong, superficial or unfair to judge beauty on the basis of physique, what's the point of merely changing the goalposts – making them wider, so to speak? You're still playing the same game. And if it's all about freedom and self [illegible]

The 'networked generation' finds TV is a turn-off

By
Consumer Affairs Editor

YOUNG adults are watching less television than they were four years ago, according to research published yesterday.

The lure of the internet, mobile phones and computer games means that those aged between [illegible] watch [illegible]

mobile phones far more than average, making 27 calls a week and sending 70 text messages. The typical Briton makes 20 calls and sends 28 texts.

They are also far more likely to listen to iPods and other portable music players, play games consoles and watch television on their com[illegible]

Digest

Saying sorry to escape jail
The "geeky" son of a millionaire who sexually assaulted a sleeping student was spared jail after agreeing to write a letter of apology to his victim. **P8**

Introducing microformats

HTML markup was always intended by its inventors to add structure and meaning to Web documents. That's cool, because that's exactly what it does. The trouble is that only about forty elements are available for you to describe the meaning of your content. Headers, paragraphs, and tables of data aren't that much of a problem; after all, that's what most of the boffins who invented the Web needed to describe. But on today's Web, forty elements just aren't enough.

Sure, you have ordered lists, unordered lists, and definition lists, but where are the elements to fully describe a book title, a review, or maybe a conversation about whether a book is any good? Microformats extends XHTML and combines all the benefits of precise meaning with greater opportunities for designers to style content using CSS. See the sidebar on page 43 for more details on microformats.

7 *Share your ideas, and collaborate with others*

Since the early days of CSS collaboration, sharing knowledge and ideas has helped moved the Web forward. Yes, I know the notion of sharing is all very "happy hippy," but it is true that most, if not all, the techniques designers now use on a daily basis have been developed by individuals who then shared that knowledge freely with the wider community.

Many of these techniques were not designed to be experimental but to help solve the common problems that designers face every day when implementing their designs with CSS.

From the earliest days of CSS layout, designers Rob Chandanais of the BlueRobot Layout Reservoir and Owen Briggs's Box Lessons shared their knowledge and their techniques. Todd Fahrner shared his findings on different browsers' rendering of font size keywords, and later A List Apart Magazine published an article by Douglas Bowman, "Sliding Doors of CSS," about transforming simple unordered lists into tabbed-style navigation.

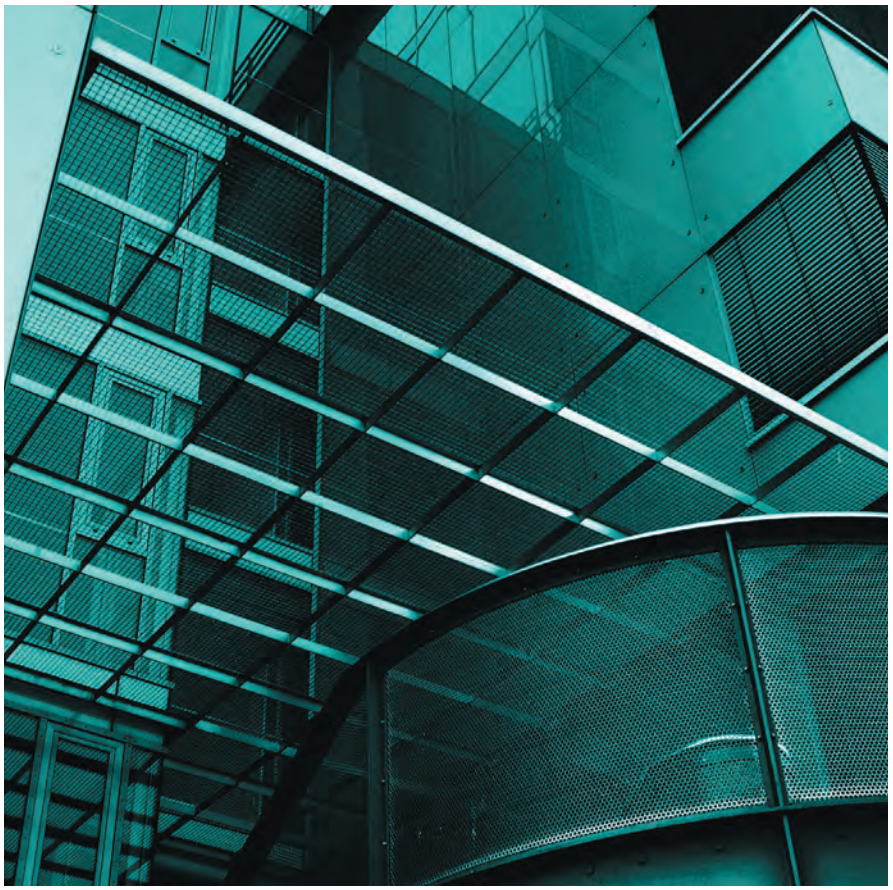
But this was not the end of designers sharing their knowledge and experiences of CSS. Since CSS began, there have been hundreds, perhaps thousands, of designers who have contribute their knowledge freely on Web sites such as A List Apart Magazine, on dedicated forums, and on their personal sites.

Whether what you share is useful for anyone but yourself rarely matters; it is the process of sharing that is valuable, and you will always get so much more from sharing an idea or

Note

An interesting use for an `id` attribute placed either on the `<html>` or on the `<body>` element is referred to as *CSS signatures*.

These CSS signatures allow visitors to make changes to the style of an individual site by adding rules to their browser's user style sheet. You can read more about CSS signatures, developed by Eric Meyer and Mark Irons, at http://archivist.incutio.com/viewlist/css-discuss/13291.



Understanding microformats

Independent designers and developers have been hard at work squeezing new meaning from XHTML. Currently, the most well known of these initiatives is *microformats*. Microformats extend existing XHTML rather than create a new language, and as such they are easy to learn and easy to implement. You need only add a set of attribute values to your markup to start using them. Let's take the rudimentary example of my business contact information. This contains a `mailto:` link, and my name will be hyperlinked to my Web site:

Andy Clarke
Principal and Lead Designer
Stuff and Nonsense Ltd.
tcss@malarkey.co.uk

You already have an `<address>` element that most appropriately describes that information, but what about elements that explicitly describe my name, my title, or the company I founded?

You can look all you want through the XHTML specifications, but I'll save you the effort. There aren't any…no name, job, or company elements—nothing. XHTML was designed to be extensible, and microformats are among the first set of extensions to be commonly adopted. Microformats use `class` and other attributes to give such precise meaning to your content.

Need an organization element? Add `class="org"`, and you have one. Need to mark up a family name and can't find an element precise enough? Add `class="family-name"`, and you just created one.

If you need to mark up any contact information so it can easily be extracted and saved in a format that can be imported into Apple iCal or Microsoft Outlook, you simply add `class="vcard"` to your `address`, and you have a simple card format that is meaningful and useful to both people and machines:

```
<address class="vcard">
<a class="url fn n" href="http://www.stuffandnonsense.
co.uk/">
<span class="given-name">Andy</span>
<span class="family-name">Clarke</span>
</a>
<span class="title">Principal and Lead Designer</span>
<a class="org" href="http://www.stuffandnonsense.co.uk/
">Stuff and Nonsense Ltd.</a>
<a class="e-mail" mailto="tcss@malarkey.co.uk"> tcss@
malarkey.co.uk</a>
</address>
```

This development is exciting. Microformats combine all the benefits of precise meaning with greater opportunities for designers to style content using CSS.

Note: You can find out more about the latest microformats and contribute to their development on the microformats wiki (http://microformats.org/wiki).

The microformats community has already released a number of new formats, with more being continually developed and proposed. New applications for microformats have already emerged, and software vendors including Microsoft are advocating for them. While being interviewed by Tim O'Reilly at Microsoft's MIX06 conference, Bill Gates said, "We need microformats."

If the Internet teaches us anything, it is that great value comes from leaving core resources in a commons, where they're free for people to build upon as they see fit.

—**LAWRENCE LESSIG**
www.lessig.org

technique than you will from keeping it to yourself. For example, I created a simple chart to help me understand specificity in CSS. To help me remember what can sometimes be a complex concept I used characters from the *Star Wars* movies. This chart was intended for my own use, but the article explaining it has since been translated into four languages. I got more satisfaction from the knowledge that the chart was useful to others than I did from making it.

Without this explosion of free and open knowledge of techniques and best practices, the use of CSS may never have grown to the level it has reached today.

Updated and better practices are being developed and shared all the time, and although CSS now has fewer technical trouble spots, new techniques are still being published that can teach you new ways to use CSS.

In October 2005, nine years after the release of the first CSS specification, "In Search Of The One True Layout" described a new method for making columns that were independent of their order in the HTML.

Although the solution is far from perfect, mostly because of browser inconsistencies and not CSS, it came as a surprise to many seasoned CSS experts. The technique clearly demonstrated we could learn new ways to use CSS, and it showed how constructive collaboration could produce an even better solution.

It has not only been caring and sharing, soft-centered individuals who have been committed to sharing their knowledge. Major corporations such as Yahoo and others have been publishing their internal libraries for public use.

The Yahoo Developer Network's UI Library of CSS tools features CSS grid templates that can make more than one hundred page layouts from a single CSS file. This sharing benefits not only the CSS design community but also those people who are sharing those ideas with others.

Despite that designers today have a much greater understanding of what CSS can do, in the future you'll see new ideas for its use and better solutions to older problems that designers haven't yet discovered.

As the last few years have shown, collaboration and sharing can be highly effective in improving our knowledge of what CSS can do. As CSS3 develops and more browsers implement parts of the many CSS3 modules, we will always have more to learn and many more opportunities to collaborate with others to create better solutions.

Note

CSS Specificity Wars is available for you to download at www.stuffandnonsense.co.uk/archives/css_specificity_wars.html

Note

Alex Robinson's "In Search of the One True Layout" is published along with many other inspiring CSS layout techniques at Position Is Everything (www.positioniseverything.net/).

Note

Many CSS templates and tools are freely available on the Yahoo Developer Network's UI Library for you to explore and implement in your own projects (http://developer.yahoo.com/yui/index.html). There is even a blog updated regularly by Yahoo developers for you to keep up with their latest ideas (www.yuiblog.com/).

What Makes Transcendent CSS Possible Now?

For many designers who are new to CSS, it might be difficult to imagine a time when creatively designed sites implemented with CSS were in the minority.

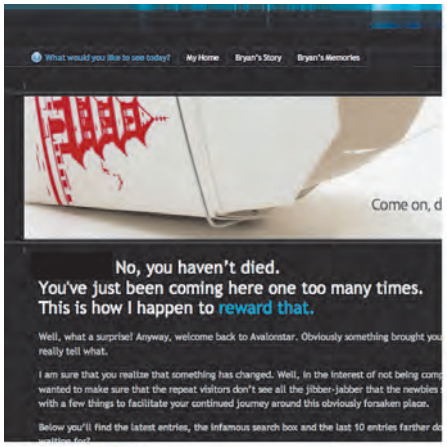
Although sites implemented with meaningful markup and CSS still occupy only a small percentage of the total number of sites launched each year, I hope we have passed the point where there can be any doubt that using CSS is not only highly desirable but also highly practical.

> *As soon as images were allowed inline in HTML documents, the Web became a new graphical design medium. Some people will just want to put out text, but some will want to apply graphical design skills and make a document. These people are, at least, a sizeable minority and there should be a means for them to achieve their ends.*
>
> —Chris Lilley (http://lists.w3.org/Archives/Public/www-html/1994May/0010.html), May 1994

In May 2003, the CSS Zen Garden's experimental playground proved that when designers work with CSS, they can create any number of exciting and inspirational designs from a single XHTML document. Outside the walls of the CSS Zen Garden, the world was slowly changing, and designers were finding new techniques to use CSS in more and more creative ways.

Note: A forerunner to the CSS Zen Garden perhaps, Microsoft's CSS Gallery (developed for Internet Explorer 3 no less) includes "Same Content, Different Style" and demonstrates just how creative CSS designers have become. Explore the CSS Gallery at www.microsoft.com/typography/css/gallery/entrance.htm.

During the earliest days of CSS when many of the solutions to everyday design problems had yet to be invented, CSS designs were thought to be boxy and boring. Thanks largely to the success of the CSS Zen Garden in appealing directly to designers, you can now visit thousands of creative CSS-based sites from the smallest of small businesses to the largest, high-profile commercial companies.


No, you haven't died.
You've just been coming here one too many times.
This is how I happen to reward that.


JEFFCROFT
A helping hand for Dvorak
Also in the blog


fortyeightdesigns
About FortyEight Designs


aftercode
claudio naboni


lgihu
a personality of james mathias
Lowlife
Art History

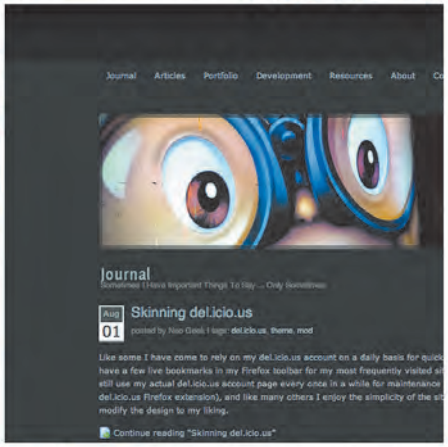

Miss Japan 2006 - Kurara Chibana


DragonArmouryCreative
LatestNews
WebCreative


GEORGE LIN
SENIOR ART DIRECTOR
Have no fear of perfection, you'll never reach it.


HOME
ARCHIVES
REVIEWS
DESIGN
ABOUT
CONTACT
BLOG
Bashing Zombies and Cleansing Ghettos
REVIEW
PREVIOUS ENTRIES


home
comics
blog
gallery
bits
about
links
contact
GALLERY HIGHLIGHT


Journal
Skinning del.icio.us


Ika Ray


Cabedge Craze
Summer '06


Jonathan Snook's Blog
Latest Entry
How to Pixelize in Fireworks
Last few entries
Finding a Shortcut


CSS REBOOT 2006
Let's see what we can
Greetings earthling


komodomed
About Komodo Media
Current happenings. Right here. Right now.

The walls of inspiring CSS galleries such as CSS Beauty and StyleGala are now full of site designs from all corners of the world and all corners of society and commerce. Regular redesign events, including CSS Reboot, encourage designers to reveal their latest redesigns on the same day, which attracts hundreds of designers, including some of the industry's best-known names (**Figure 1.20**).

Unexpected uses for CSS

Over the past few years, CSS has been cropping up in all manner of unexpected places, from instant messaging to everyday applications, such as Web browsers and e-mail clients, right down to the desktop.

In fact, Adium (http://adiumx.com) is an alternative chat client to iChat AV for Mac users that offers them the ability to choose between hundreds of interface themes, all downloadable from its Adium Xtras site. Whether you prefer your chat windows to look like metal, look like shiny plastic, or perhaps even resemble a terminal window (if you want to feel really hardcore), the options should keep you entertained for hours.

But the fun doesn't stop there, because beneath the surface of Adium is a chat window made from XHTML and message themes that are styled using—guess what?—CSS. So if you, like me, can't live without a chat window emblazoned with targets, arrows, and the Union Flag, you can easily create new themes and pass them around to your friends. I'm sure they will all love you for it.

CSS is also a key component in the look and feel of Mozilla applications, including the Thunderbird mail client and the Firefox browser. In these applications, you can style buttons, windows, pages, menus, and sliders all using CSS and images.

The Apple Dashboard Widgets are also created using a combination of XHTML, JavaScript, and CSS as well as some proprietary Apple Script. This simple mix of Web technologies makes it easier to develop new Dashboard Widgets or to edit the thousands of others that have already been developed.

Achieving camaraderie in the CSS community

Support does not mean that everybody gets the same thing. Expecting two users using different browser software to have an identical experience fails to embrace or acknowledge the heterogeneous essence of the Web. In fact, requiring the same experience for all users creates a barrier to participation. Availability and accessibility of content should be our key priority.

—NATE KOECHLEY
http://developer.yahoo.com/yui/articles/gbs/gbs.html

Graded browser support

Until 2006, MOSe and progressive enhancement were topics largely discussed only in relation to blogs or experimental designs and never in relation to large-scale, commercial Web sites. Many designers thought the approach was too risky while the majority of Web users still browsed using Internet Explorer 6 for Windows.

Surprisingly, affirmation that the techniques central to the Transcendent CSS approach are practical on large-scale commercial projects came not from discussions on designer forums or blogs, but from one of the Web's true giants—Yahoo.

In February 2006, Yahoo standards evangelist Nate Koechley published a Yahoo Developer Network document called "Graded Browser Support."

What exactly does support mean?

Koechley's article makes it clear that it is neither possible nor desirable for people accessing Web content using different browsing technologies or devices to expect to receive exactly the same design. After all, a person will have a different experience browsing the Web using a large desktop monitor than someone using the small screen of a handheld PDA (personal digital assistant) or mobile phone. Extending that notion to browser versions is only a small step.

Graded browser support does not exclude users of older browsers from accessing content; it simply acknowledges that not all visitors will see the same levels of visual design. Koechley explains the concept:

> *An appropriate support strategy allows every user to consume as much visual and interactive richness as their environment can support. This approach builds a rich experience on top of an accessible core, without compromising that core.*
>
> —Nate Koechley, (http://developer.yahoo.com/yui/articles/gbs/gbs.html)

Note

Although it is now clearly showing its age, the BBC Browser Support Standards table, available at www.bbc.co.uk/guidelines/newmedia/technical/browser_support.shtml, is still an interesting read.

Browser grading

In the past, the widespread adoption of "advanced" CSS has been stymied by the view that a design should look the same across all browsers and platforms. To help solve this problem, Yahoo grouped browsers into three "grades": C-grade, A-grade, and X-grade.

Table 1.2 Yahoo-Graded Browser Matrix

Browser grades	Description
C-grade	Visitors using "incapable, antiquated, and rare" C-grade browsers experience a basic level that consists of core content and functionality. The content and experience is "highly accessible, unenhanced by decoration or advanced functionality." Layers of style and behavior are omitted.
A-grade	Visitors using A-grade browsers can take full advantage of their browser's "powerful capabilities of modern Web standards; the A-grade experience provides advanced functionality and visual fidelity."
X-grade	X-grade browsers include "fringe or rare browsers." Browsers receiving X-grade support are assumed to be capable and not "choke on modern methodologies."

Yahoo browser grading charts are updated approximately every quarter. This chart is as of August 2006.

Browser support standards

The idea of compiling a table of supported browsers is of course not new with Yahoo!. Many large organizations and content providers such as the BBC long ago developed what have often been called *browser matrixes*, and guidelines, for their developers in relation to "target," "supported," and "unsupported" browsers. However, the BBC's approach still harks back to the notion that the most popular browser of the day should be the target browser irrespective of its capabilities.

What is different about the approaches taken by BBC and Yahoo! is Yahoo!'s commitment to the notion that it is acceptable to provide different levels of design experience for modern rather than older browsers and that a target browser need not necessarily be the most popular. In taking this transcendent approach, Yahoo! has demonstrated and stated publicly that Transcendent CSS use is not only possible but required if the Web is to move forward quickly, not stagnate or move at the pace of the slowest browser.

Thousands of designers and developers from all over the world have adopted Web standards. This is great news for anyone who is either creating or consuming Web content. Finally, it is possible to use a Transcendent CSS approach with confidence and know you are in good company.

Discovery, process, inspiration, and transcendence

Transcendent CSS is part of a larger process that involves the following:

- Evaluating how you think about and use markup and CSS
- Thinking about the way you work and how you collaborate with others
- Reconsidering old-fashioned ideas about cross-browser compatibility
- Positioning meaningful, semantic markup at the center of everything you create

Knowing this, it's now time for you to put on your parka, kick start your scooter, and move off. In this book, you will ride pillion through all the important facets of Transcendent CSS. Don't worry, you don't need to put your arms around me; we don't know each other well enough yet!

You will start at the beginning with the **discovery** of the *content-out* approach to using markup, and you'll learn how you should always structure and order markup meaningfully rather than according to how it looks or its visual layout.

Then, in Part 2, you will learn to take a new perspective on the **process** of Web design, finding out more about the meaning-based workflow. You will see how to combine visual design and meaningful markup earlier in your design process by working with standards-based prototyping. If by then you want to get your hands dirty, you will get to build a standards-based prototype using XHTML, CSS, and a little JavaScript.

You will then look for design **inspiration** (Part 3), paying attention to grids and how they have been used in Web design. You will learn how you can derive alternative types of layouts from grids that have been inspired from other media. It is here that you'll learn to look around you for more unusual forms of inspiration to bring home to your designs.

Finally, you will learn how all this fits together in **transcendence** (Part 4) by working through examples that show you how to accomplish Transcendent CSS techniques. You will work with all kinds of positioning, floats, and other layout techniques; learn new techniques for using advanced CSS2.1 selectors; and look at working with different media types.

And if that isn't enough, you will finish by learning about CSS3, which is the next version of CSS, and the many exiting creative opportunities it will offer. You will even get to see the CSS3 Advanced Layout Module and can download all the example files to see it working in action.

Designing from the Content Out

Way back in the mists of 1997, typographer David Siegel changed the Web, as we knew it, when he wrote about an emerging technique for laying out Web pages. Somewhere in the dim light of a laboratory, an HTML `<table>` element had been stitched together with a spacer GIF and then flooded with 10,000 volts of electricity. This gave life to the idea of using `<table>` markup as a means for visually laying out pages.

Designers rejoiced that they had found a way to reproduce some of traditional print media's layout conventions on the Web, caring little, if they even realized, that the inventor of the `<table>`, Dave Raggett, had originally intended his creation to present tabular information.

But while Web designers partied like it was 1997, Siegel soon recognized he had unleashed a monster. Not long after, he wrote "The Web Is Ruined and I Ruined It":

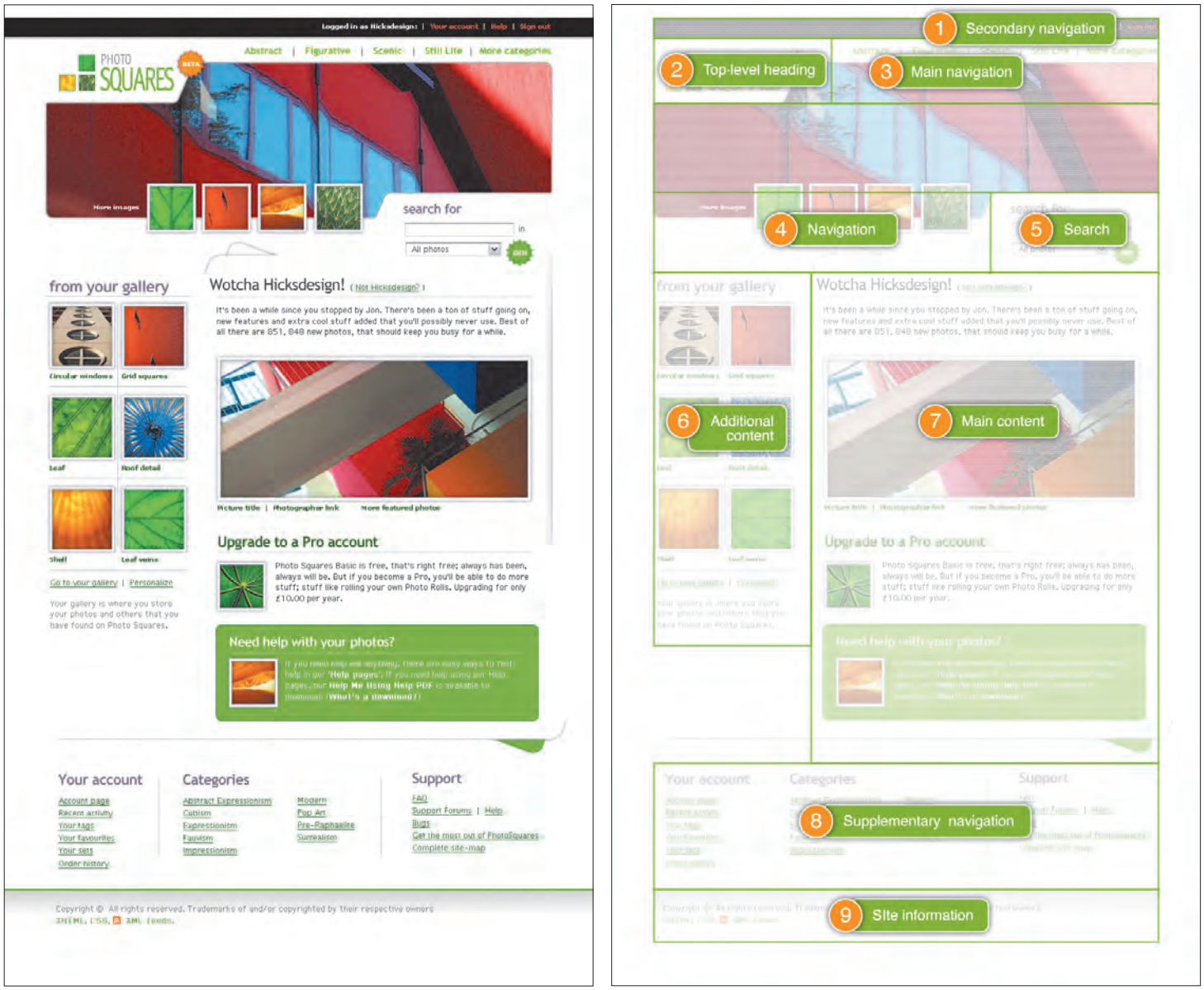
> *Some people say I've ruined the Web, and to them it's true. [...] I ruined the Web by mixing chocolate and peanut butter so they could never become unmixed. I committed the hangable offence of mixing structure with presentation.*
>
> —David Siegel (www.xml.com/pub/a/w3j/s1.people.html)

Not only did Siegel's monster go on to spawn several "killer" sequels and millions of Web pages that mixed up their content, structure, and presentation, it also reinforced the thinking that content and structure should depend on visual layout in the minds of Web designers. This notion has stayed in the minds of Web designers ever since to the detriment of flexibility, semantic meaning, and accessibility.

Whereas we understand now that it is always preferable that content and structure be independent from visual presentation, table-based layouts impose *their* order on the content of a document to achieve a specific visual result.

The content order of a table-based design may make perfect sense to a visitor who can see the visual layout, but taking away that visual presentation and that order can make the content incomprehensible (**Figure 1.21**).

Therefore, many people, including Siegel, soon advocated against using tables for layout. But in the absence of a viable, working alternative, the monster was left free to roam.



1.21 **Left: View of the Web site. Right: Removing the visual presentation makes the content order incomprehensible**

Fortunately, we now have CSS to hunt down and destroy the beast. But even though CSS layouts are now possible in almost every situation, designers continue to find it hard to move away from presentational thinking about the structure and order of their content.

The content-out approach

Peel back the skin of many modern CSS layouts, and you will find that presentational markup and content order still remain. CSS-styled pages are often still constructed in a "top-to-bottom, left-to-right" order that has been designed to satisfy the cravings of the visual design. But this time they use `<div>` elements rather than a `<table>` element.

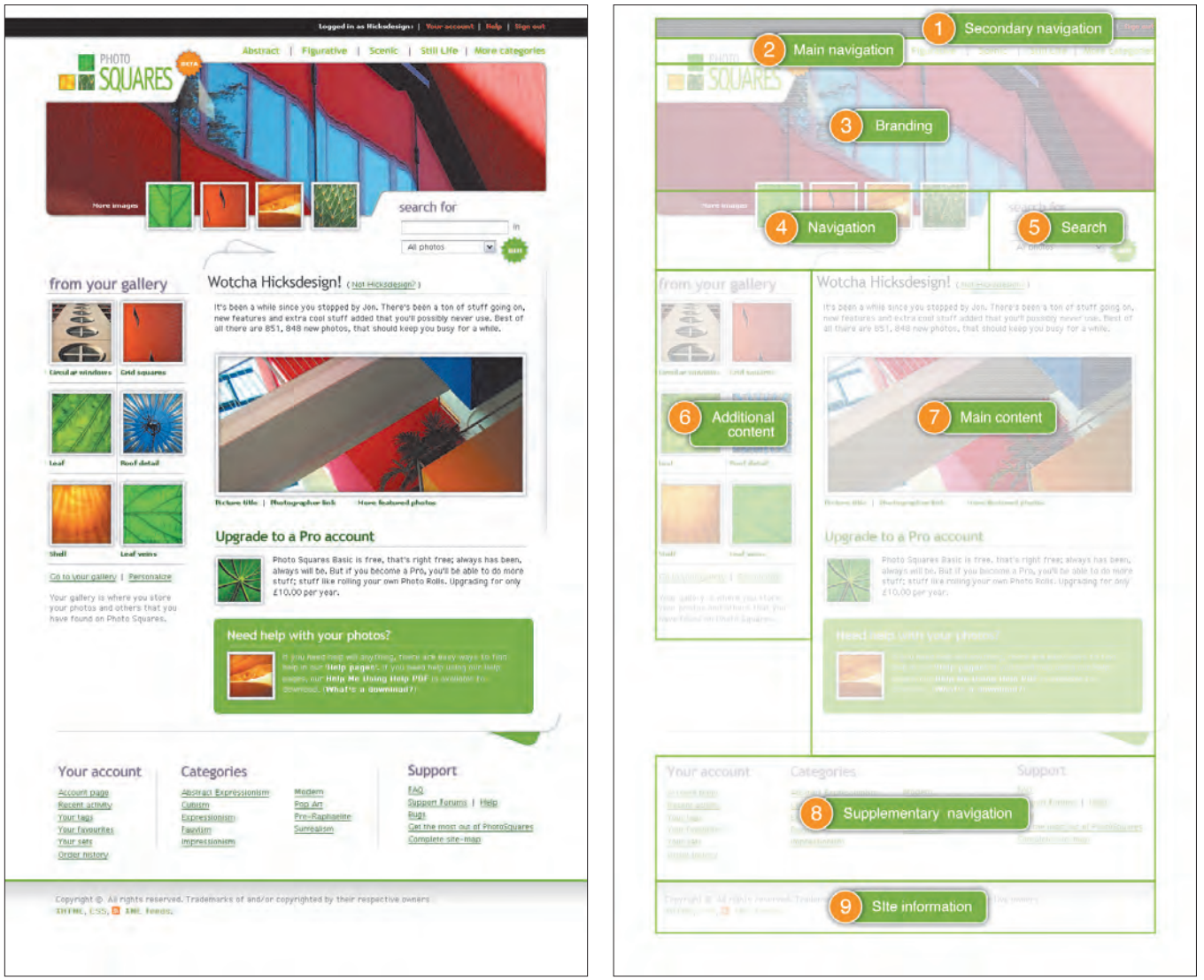
The HTML Working Group at the W3C originally intended the `<div>` element to be used for the semantic grouping of areas with related content. But many designers now use them in the same way they used tables, to achieve a visual layout, without paying much attention to their divisions' semantic value.

Simply replacing `<table>` cells with `<div>` elements will not help you gain the full benefits of using Web standards or CSS. Unless you have carefully considered the meaning of each division, `<div>` elements are little better than using tables, particularly when they are nested several levels deep. Unfortunately, although they have continued to improve the quality of the code they output, visual Web editors such as Dreamweaver continue to create code that contains an abundance of nested divisions. By adding more `<div>` elements than are absolutely necessary, not only do you increase the size of your documents, but you also increase the likelihood of errors creeping in during your development.



A typical, nonoptimized CSS layout

Consider the content order of a simple but typical page that has been styled with CSS (**Figure 1.23**). This document contains branding, two sets of navigation links, two related areas of content, and an area holding site-related information.



1.23 **Examining the typical content order of a CSS layout**

Note

The Linearize Page feature of the Web Developer extension for Firefox (https://addons.mozilla.org/firefox/60) is a valuable tool for showing the order of content within any Web page.

Remove the CSS on many sites using this visual structure, and you will see that the content order typically runs something like this:

```
<div id="branding">Top-level heading</div>
<div id="nav_main">Main navigation</div>
<div id="content_sub">Secondary content (left)</div>
<div id="content_main">Main content (right)</div>
<div id="nav_sub">Secondary navigation</div>
<div id="site_info">Legal and copyright information</div>
```

Here the order of the content follows a table-based layout, and in essence the designer has done little more than replace table cells with semantically named divisions.

Optimize the content order with or without styles

Looking at the previous example with no style sheet attached, you will see you could make many improvements to the order. The Web Content Accessibility Guideline 1.0 specification clearly states what you should do:

> *(6.1) Organize documents so they may be read without style sheets. For example, when an HTML document is rendered without associated style sheets, it must still be possible to read the document. [Priority 1]. When content is organized logically, it will be rendered in a meaningful order when style sheets are turned off or not supported.*
>
> —WCAG 1.0 specification (www.w3.org/TR/WAI-WEBCONTENT/)

Another important issue, of course, is not only content order, but navigation order.



Navigation

The two areas of navigation are related but kept separate, one near the top of the source order and the other near the bottom. Sharing a common parent `<div>` and having their own identities would subtly add more semantic meaning to both navigation lists:

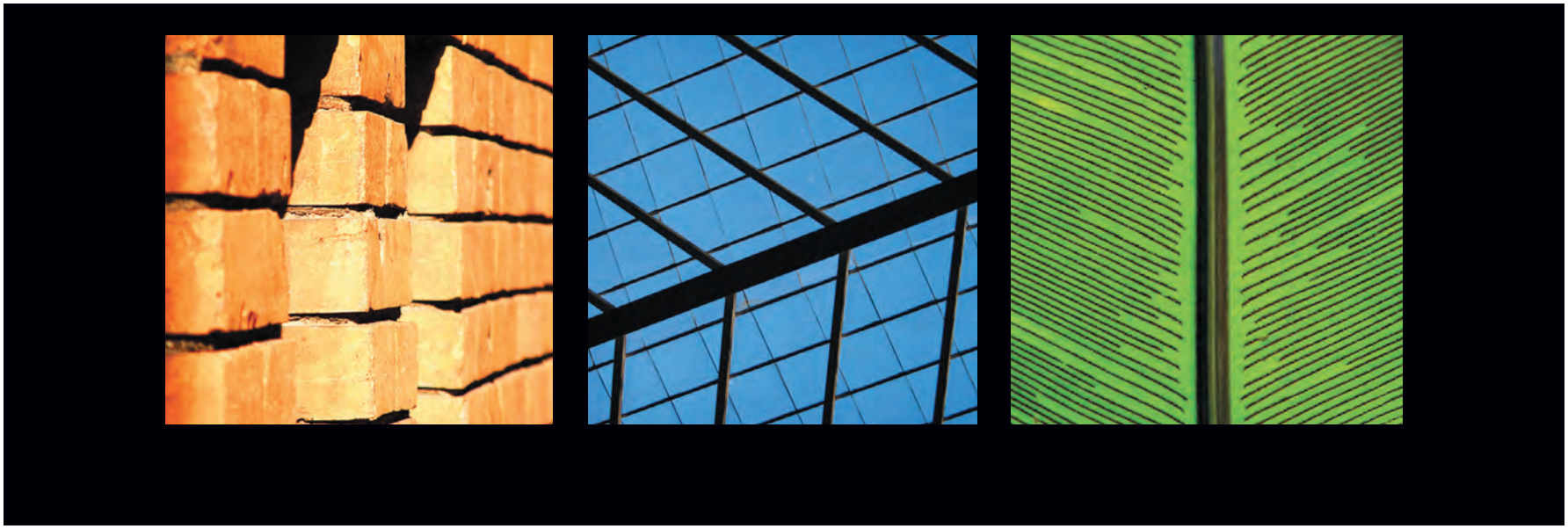
```
<div id="nav">
<ul id="nav_main">
<li>Main navigation</li>
</ul>

<ul id="nav_sub">
<li>Secondary navigation</li>
</ul>
</div>
```

You could then place *both* these navigation lists either near the beginning or near the end of your document source and use CSS positioning to place them visually wherever you require for your design.

Branding and content

You could write the two related content areas in the order they appear onscreen and not in the order that makes sense without styles.



You will also see that the branding area, which typically will contain a top-level heading, is separated from the content that follows. Actually, in this example, the heading does not require a containing , because you can apply styles to the heading to create the same visual result:

```
<h1>Top-level heading</h1>
```

Swapping the order of the content elements and placing the heading directly before them restores the relationship between the heading and content:

```
<h1>Top-level heading</h1>
<div id="content_main">Main content</div>
<div id="content_sub">Secondary content</div>
```

This revised order will better suit the needs of visitors who cannot see the visual layout (**Figure 1.24**):

```
<h1>Top-level heading</h1>
<div id="content_main">Main content (right)</div>
<div id="content_sub">Secondary content (left)</div>
<div id="nav">
<ul id="nav_main">
<li>Main navigation</li>
</ul>
<ul id="nav_sub">
<li>Secondary navigation</li>
</ul>
</div>
<div id="site_info">Legal and copyright information</div>
```

Using CSS, you have greater control over layout than was ever possible with table-based designs. You are largely free to change the visual layout of a page without altering its linear order. This greatly improves accessibility for people who access Web content using "linear browsers" such as screen readers. When CSS styles are removed or not available, what remains is well-ordered content.

With the positioning tools offered by CSS and an increased knowledge of how you can use them effectively, it is time to finally let presentational markup sleep the long sleep. You can help it rest in peace by learning to adopt a content-out approach.

1.24 **Left: The most appropriate order for the content, 1 through 9.**
Right: That order superimposed on the visual design.

adactio

{Skip navigation}

- Journal
- Portfolio
- Articles
- About
- Contact

Wednesday, April 5th, 200

Naked

My site has no stylesheets ("how does it st

In any case, it's nice just to look at the flow enabled, the visual layout makes this headi

The man responsible for this CSS nakedne

I had a pleasant chat with Dustin yesterday listen back to. I ramble on about JavaScript

11:40am

Tuesday, April 4th, 2006

When mashups attack

In all the many mashups out there, Google

One of the latest in the long line of map mi

Has Nick Denton gone too far this time? G Jesse are just so excited that George Cloon

9:50pm

Sunday, April 2nd, 2006

Upcoming webolution

At the risk of becoming API-watch Central

Aaby and the gang have been diligently ge being exposed through the API. Methinks

On the Upcoming site itself, you can now l few days after Brian Suda mentioned in a c during XTech).

Feature wished for; feature added. This is passionate users. By all accounts, her panel days. Muggins here missed it but I have a g

11:14pm

Friday, March 31st, 2006

Prime yourself

Last year, there was a lot of positive hum

I rented and watched the movie last night. about the movie. Now I'm beginning to ge

I would like to offer some humble opinion

1. Try to not read or hear *anything* abou
2. Watch it. It's great. It looks great, it s
3. After watching it — **but not before** diagram.
4. Now watch it again.

You may want to just go ahead and buy th

11:02am

Monday, March 27th, 200

Simple Storage Service

Amazon's new S3 service looks very intere the really fascinating aspect is that there is

> When you get down to it, Amazon S useless.

The creators of S3 have gone out of their w networks.

Leaving aside the underlying technology, S barrier to entry has been lowered for anyon nature of S3 means that you can store and s

Hardware has been getting cheaper and ch bandwidth issues, be sure to listen to Bern

I'm looking forward to playing around wit

11:03pm

Sunday, March 26th, 2006

Darwinian webolution

Odeo have released an embedded recorder

Del.icio.us now offers private bookmarks.

Flickr now marks up profiles using the hC

adactio / Jeremy Keith

Source of http://flickr.com/p

Something that became very clear — both

Skip to content Skip to subnavigation

Focal Curve

- Info
- Work
- Blog

Where's the design?

This site is participating in "Naked Day", Apri pure structural HTML displayed with your bro http://naked.dustindiaz.com/.

Going Naked

April 4th, 2006 :: posted in General Geekery

Last week Dustin Diaz proposed the First An separation of content from presentation. A wel

So, on a lark and at the last minute, I jumped worked out, eh? If you're reading this in the a in your browser of choice. If you're seeing thi SFW nudity.

Focal Curve

- Info
- Work
- Blog

Where's the design?

Going Naked

No Comments Yet »

Rejoining the Cult

March 31st, 2006 :: posted in General Geekery

Well I finally did it. I bought a Mac. A 15.4" and my very first Apple notebook. I've had it thing ever to rest in my lap. Well, technology-

I grew up with Apple computers. In the fourth family Apple IIGS, graduated to a Macintosh 6116CD my sophomore year.

In 1999 I jumped ship to a Windows PC, with of the cost of a new Mac. The Performa was g rag-ing 500MHz Celeron (assembled by my fr Windows 98. That brandless black box has ses with a failed power supply, though I really mu must recover.

I acquired a Toshiba Satellite A15 notebook i Thanksgiving riot/sale. It's been my primary w to an outlet until I round up a replacement batt

But all along I've intended to someday return Bay Street and walked out as a new man. A N was out of the box and in my hands the real ev

Outside

I was immediately stricken by the beauty of th and seamless aluminum shell. The bottom surf bulges). The interior is a broad flat plane, with whitespace. The MacBook just *feels* good to h lid tenderly when I put it to sleep.

Many small details are noteworthy as well. Th out to be a spiffy little innovation. The magnet just as it's intended to do. The built-in iSight c ghostly backlit keyboard adjusts automatically grilles, and I enjoy covering and uncovering p wee button of machined metal and a fine strea for.

Inside

My sum prior experience with OS X consisted my workflow/lifestyle in regular productive us downloading all the vital software and finding

I'm sticking with Firefox as my browser of ch regularly using Safari, Camino and OmniWeb messaging, since I talk to a few people outside MAMP made setting up a local dev server a b

As far as serious productivity, I have yet to do TextMate vs skEdit... TextMate is in the lead) performance under Rosetta emulation until un survival as water and oxygen]. And while ther Word:Mac.

Having used OS X for a solid week now, I'm system than Windows. The machine seems to

The Airing of Grievances

Lest you think me a gushing fanboy, I do have become uncomfortably hot to the touch after p casing's heatsink effect to provide enough coo seems worth it aside from the risk of scorched chassis while lifting it up to allow better air cir

As with the PowerBook, all the ports are locat widened, taking up more desk space (the plug

My only other complaints thus far aren't flaws keyboard (I could plug one in, but most PC no of a long line of code. There may be an equiva keys behave differently in OS X, but they cou something I'll have to get used to.

joshuaink

"more love than a bus full of hippies"

Naked day

Wednesday April 5, 2006

I didn't officially register for it but I am doing I will save! Enjoy!

- Skip to main navigation
- Skip to sub content and search form

Gnarls Barkley's Crazy for

Tuesday April 4, 2006

So, on the offical Gnarls Barkley site it seems

Am I pissed off? Well yeah. I just spent 79 pc

http://www.gnarlsbarkley.com/wuzio/crazy.mp

- Skip to main navigation
- Skip to sub content and search form

Announcing Moral outrage

Tuesday April 4, 2006

You know me, I like to set up these little gimm

In light of recent events I thought I should stan provoke moral outrage, all over the world. To post.

So, watch out this Monday, for the first (and p

- Skip to main navigation
- Skip to sub content and search form

Another big pair of tits

Tuesday April 4, 2006

Well the survey says, yes.. well... no, well yo

So instead, let's do it again, another big pair o

Now, I don't know about you but I find these

.. Fool me once, shame on... shame on

Something to say? Do tell [9]

- Skip to main navigation
- Skip to sub content and search form

LiveSearch:

Style options and feedback

- Change the look and feel of this site

Recently

The last seven posts made on this site.

- Naked day
- Gnarls Barkley's Crazy for free?
- Announcing Moral outrage Monday
- Another big pair of tits
- A quick survey
- Thinking out loud about my career and
- R3c0gNiz3 my $/ci11$

del.icio.us

- www.ajaxwrite.com
- misprinted type 3.0 - art, design and typ
- Code Style
- kitsimons - Simon Kitson's website
- OOP Concept explained: Polymorphism
- Ladystmers
- The (real-life) Simpsons
- Elastic Water on the ISS
- Web 2.0 Generator
- Mouseover DOM Inspector
- Le Vase
- RSSContact
- RSSCalendar.com
- Sitemap Thing
- Linx

Geeky ~~shite~~Meta

- Subscribe to this site
- Gvisit visitor logs

A PROUD MEMBER

Main navigation

Navigation for every section of this site

- Skip to sub content and search form
- Top of the page

- Blog
- Archives
- Kitchen
- About & Contact

Page foot

- Top of the page
- Skip to main navigation

Simon Willison's Wel

PHP, Python, CSS, XML and general web de

Skip to Navigation

- **Blogmarks:**
- Boot Camp
- JavaScript apps with read/write access t
- gotoAndLearn.com
- Code Elegance, Code Balance
- YouTube - danah on O'Reilly Factor
- Xara (vector graphics programme) goes
- Hex Fiend
- Blown away (again) by Hack Day
- Styles: Beyond WS and REST
- favcol (via)
- Backing Up Flickr Photos with Amazon
- BitBucket - Experimenting with Amazo
- Amazon S3
- Friendster lost steam. Is MySpace just a
- Form Hijack (via)
- ... and 1969 more [RSS]

[16 hours, 37 minutes ago] Naked d

Naked and proud.

[5th April 2006 - 00:09 | Comments (5) | Ping

[6 days, 5 hours ago] Learning Flas

I've decided it's about time I learnt some Flash now that Flash is the most practical option for JavaScript as well. Google Finance and the Y

I have minimal design skills, so much of the Fl Flash aimed at programmers? Something simil welcome.

[30th March 2006 - 10:53 | Comments (19) |

[4 weeks, 1 day ago] My ETech Jav

I gave a three hour JavaScript tutorial at ETech as a programming language. It seemed to go p found it interesting.

I didn't finish the presentation in time to get ha the slides here. The 111 slides (reduced to low

For the sake of completeness, I'm also releasin very closely but they might be of interest in an

The photos in the presentation were all found

Update: I've converted the notes to a single H

There's a lot of interest in the slides as a single mean-time, I've posted higher quality copies of

[7th March 2006 - 05:42 | Comments (30) |

[1 month, 2 weeks ago] Yahoo! UI

The Yahoo! Developer Network was updated User Interface Blog.

Here are some of the highlights:

- Mature, extensively tested cross-browse
- CSS-skinnable UI controls built on thos
- A library of documented design patterns
- A description of Yahoo!'s Graded Brow HTML, accessibility, progressive enhan

The code is all under a BSD Open Source lice

This release represents the culmination of mor to the work that has gone in to the code.

I've been playing with this internally quite a bi

- The Event library normalises the event with Safari. It cleans up events when th an event to the DOM ID of an element finishes loading.
- The CustomEvent library implements th
- The Dom library's getStyle and setStyle
- The Animation library uses a global An on how much stuff is happening - so if is going on at the same time.
- Here's a surprisingly tricky question: wh The Drag and Drop library defaults to st
- The Drag library also makes it easy to h example.

And finally, I couldn't resist quoting this

```
var attributes = {
    width: { to: 100 },
    height: { to: 100 },
    fontSize: { from: 100, t
    opacity: { to: 0.5 }
}
var anim = new YAHOO.util.A
  YAHOO.util.Easing.easeOut
```

Source

A big thanks to the team that put this all t

[14th February 2006 - 14:01 | Comment

[1 month, 3 weeks ago] Notes fr

I'm at the Carson Workshops Future of V Here are the notes I've gathered over the

- Joshua Schachter
- Cal Henderson
- Tom Coates
- David Heinemeier Hansson
- Shaun Inman

What happened to the design?

To know more about why styles are disabled

MovableType to Wor

Apr 03

18

I've done it: as of 10 minutes ago, and thanks WordPress, and leave MovableType behind.

It wasn't an easy decision: I've been toying w for change (though there were reasons, believe

(more...)

continue reading...

Asian Persuasion

Mar 30

none

Let's turn back the clock a little, shall we? A discovered that the late-2004/most of 2005 des XHTML/CSS. In fact, from the screen shots

(more...)

continue reading...

- Homepage
- Entry Archives

- Homepage
- Site archives
- About this site
- Get in touch

- MovableType to WordPress
- Asian Persuasion
- Word of the Day
- SxSW 2006 Day 5
- SxSW 2006 Day 4
- SxSW 2006 Day 3

more entries...

- Official Blogger Template
- Navigation Matrix (CSS)
- css Zen Garden #24 (CSS)
- View Browser Source (OS X)
- Sony Clié Icons (OS X/WinXP)

- Budget Design (PDF)
- Cascading Style Sheets (book)
- The Zen of CSS Design (book)

Primary Feeds: Full Text | Comments

Secondary Feeds: Ephemera | Blogroll

What are these links?

Textmate, Transmit, Photoshop, Illustrator, W

Copyright ©2001-2006 Dan Rubin All rights

Work Inquiries? Contact me... I'm available fo

Dan Rubin's SuperfluousBanter [ISSN 1543-

Web Typography
typographic style for the web

Zeldman: Unmixed
why he's still the man

SimpleBits, Arkanoid Edition
Dan Cederholm realigns

Veerle's blog 2.0
pure design perfection

CNN redesigns
3 columns, other goodies

Standardista Table Sorting
sort your tables with javascript

more ephemera...

garretdimon.com
dig his snazzy IA Templates

hivelogic.com
Benjamin. Dan Benjamin.

daringfireball.net
dude, it's John Gruber.

nundroo.com
Didier Hilhorst, private eye

andybudd.com
bloody english...

zeldman.com
all hail the z-man

more links...

What happened to the design?

skip nav

Site contents

- Blog
- Blog archive
- Blogmarks
- Music
- Photos
- Sandbox
- Referrers
- About
- Accessibility
- Search

CSS Naked Day posted 1 day

- Today, April 5th, is CSS Naked Day. T CSS stripped off. (Technically you are

So why CSS Naked Day? The original semantic, meaningful markup. Even the

> This is a fun idea, fully in line wit presentational capabilities, saving people who participated in the be

So is your site laid bare? Is it site still un

Web standards · CSS techniques · Mark-up le 14 comments [add/view/link]

SxSW wrap up posted 2 week

- Yes I know I've been back for nearly a it any good? Abso-bloody-lutely. The i bunch more folks who I missed last time handful). Bizarrely I also got to meet so as with last year, I shared drunken conv that front last year, but this year we boo breakfast, free wifi, free beer and the bl

But what of the conference itself? It wa last year. It seemed the majority of the a BlogHer. The increased attendance also conversational feel provided a by a smal

Overall I didn't feel quite as excited by really were inspiring, fun and/or fascina odd brief note through some of the pane

Traditional Design and New T

Mark Boulton waxed lyrical about Phil

Book Digitization and the Rev

Really fascinating panel on a subject ab The interesting thing here is that both co commercial application, that of supplyin copyright and access issues involved.

How to Make the Most of Map

Well I had to attend this didn't I, and it show off the stuff they've put together. API, there are Community Walk and W on Wayfaring are the Jackracker which the origins of fine cheese in France (ther

Rev Dan Catt, formerly of Geobloggers website such as Placeopedia.

Web 2.1: Making Web 2.0 Acc

Not as much practical advise here as I'd make on the progress and applicability Don't even read 'Understanding WCA formats (HTML, Flash, PDF, etc) and a and content producers what they practi test. Which is easier said then done to be centred design". We might use that one.

It was also good to hear ATAG mentio comments, etc.

How to Roll Your Own Web C

In true capitalist style Jason Fried says t things slightly differently for An Event thanks to SxSW we've got some great s

Jason also mentioned that the Gleacher a fascinating and potentially very useful

Web Standards and Search En

An entertaining panel, especially for wa to this attention, Mayer probably had th

Mayer talked about what Yahoo! looks across what it is about. Mayer intimated the websites with 'quality' content also validate are likely to have good content, Yahoo!'s job to promote web standards documents than validation per se.

Mayer also intimated that accessibility c accessible sites"). Of course this would passes for example are not an accurate

Mayer introduced many of the audience pages are indexed, which have links in

On the subject of rel="no-follow", M credit the link. Mayer also said that, of c actually just doing poor SEO.

Microformats: Evolving the W

Another engaging panel, especially as I already started on this) so that independ

I also got introduced to Tails, a Greasem sort of thing Flock might do in the futur

What happened to the design?

Zur Navigation springen , Zum Inhalt springen

1. 01 Start
2. 02 News
3. 03 Artikel
4. 04 Forum
5. Volltexisuche

05 Suche

Schrift: grösser • kleiner Stil: Druckansicht •

Navigation:

- accessBlog
- Suche
- Hilfe
- Blogroll
- Archiv
- News abonnieren: RSS

Tags? RSS? News?

Wir erklären Ihnen, was das heisst und wie Sie

Tags:

- AJAX
- ATAG
- Ausbildung
- Barrierefreiheit
- **Barrieren**
- **Best Practice**
- **BIENE**
- **BITV**
- **Blogs**
- **Browser**
- CMS
- **CSS**
- **Design**
- DOS
- eCommerce
- eGovernment
- eLearning
- **Flash**
- **Gesetze**
- Hausmitteilung
- **Hilfsmittel**
- **HTML**
- **JavaScript**
- **Leichte Sprache**
- Lernbehinderung
- **Linux**
- Literatur
- **Mac**
- Mobile Web
- Multimedia
- Navigation
- Österreich
- PDF
- Schweiz
- Schbehinderung
- **Testen**
- Typografie
- UAAG
- **Usability**
- **Veranstaltungen**
- W3C
- **WCAG**
- Web 2.0
- **Webstandards**
- **Werkzeuge**
- **Windows**
- **Zertifizierung**

Kommentare im Blog:

- *Jens Meiert:*»EfA nackt!...«
- *Weisshart:*»EfA nackt!...«
- *macx:*»Webstandards zum mithören...«
- *Ansgar Hein:*»Betrachtungen zu PAS 7
- *mobily:*»Wochenendbeilage...«

accessBlog:

04 Apr 2006

EfA nackt!

Wo ist denn unser Design hin? Ganz einfach, man sämtliche Formatierungen abschaltet? Ve

Entstanden ist die Idee im Weblog von Dustin Liste von Websites, die ebenfalls an der Aktio Verzeichnis umbenennen.

Nachtrag: auch wenn heute erst der 4. April dieser Zustand vorbei ist.

Kommentare: 2, Permalink
Tags: CSS, HTML

03 Apr 2006

Liebe Länder: Cool URIs don't cha

Hausmitteilung: in unserer Serie zum Stand de verlinken. Und regelmäßig lassen wir einen L dabei fest, dass wieder ein paar Landesgesetze (Vielen Dank an Nina Gerling, die sich die M

Kommentare: 0, Permalink
Tags: Barrierefreiheit, Gesetze, Hausmitteilun

31 Mär 2006

Fixpert :: web design

The World of Fixpert

- My Portfolio
- Bike Lane
- Gadgets & Technology
- Pop Culture
- Random
- Web Design

Flickspert

Links to the Homies

- XAPRB
- i thunk so
- Waldo Jaquith
- Nowhere Skateboards
- Barf Comics
- Brain on Fire
- Rick Whittington Consulting

If Fixpert Looks Bare...

That's because it's CSS Naked Day. I feel sc is baring nothing but pure xhtml markup for

» Continue reading

Comments (0)

Posted April 5th, 2006 in Web Design

Diggnation Got Me Throug

This is a story about how Diggnation, one of

» Continue reading

Comments (2)

Posted April 4th, 2006 in Random

No Etech 2006 Podcasts?

Etech 2006 — wherefor art thou '06 coverag

» Continue reading

Comments (3)

Posted March 30th, 2006 in Web Design

Caught on Film: Driver Att

What a bastard. This makes me so mad.

» Continue reading

Comments (4)

Posted March 21st, 2006 in Bike Lane

Wild Pit Bulls...Couldn't D

My neighbor's pit bull had puppies that are n

(These aren't the pups in question, but aren't

» Continue reading

Comments (5)

Posted March 20th, 2006 in Random

Hello to my SXSW Friends

Hi everybody. I'm back from my trip to Aust Winter is still going on here.

» Continue reading

Comments (8)

Posted March 17th, 2006 in Web Design

Trusted Filters, del.icio.us,

For better or worse, lack of good content and the drain checking my RSS feeds and postin

» Continue reading

Comments (4)

Posted February 24th, 2006 in Web Design

In Your Face Snowboardin

At long last. Remember back in January whe

» Continue reading

Semantics Is Meaning

In technical terms, *meaning* is often also described as *semantics* and has become a hot topic among designers and developers. Even among those who have been working with markup for a while, disagreements still often occur over choosing the most appropriate markup to add the most fitting meaning. Molly E. Holzschlag sums this up nicely:

> *In markup, semantics is concerned with the meaning of an element and how that element describes the content it contains.*
>
> —Molly E. Holzschlag (www.informit.com/articles/article.asp?p=369225&rl=1)

CSS Naked Day

A world (wide Web) without style might seem an odd concept, but increasingly many designers have been considering how their pages would behave in such a world.

In an attempt to encourage designers to focus on the "naked" structure behind their visual designs, Dustin Diaz proposed removing the CSS style sheets from their sites for one day (**facing page**).

Diaz's aim was to highlight the use of meaningful markup and the importance of structure and content ordering. Along the way, it also confused a good many visitors, who no doubt thought their browser was broken when many sites that day appeared in only the default browser styles.

CSS Naked Day illustrated the need for designers to structure and order content logically before beginning any work on accomplishing a visual design.

It clearly showed that with no distracting visual layout, the meaningful structure of your naked content becomes clear: Visitors can more easily see headings and hierarchy, and they can more easily identify paragraphs, quotations, and lists.

Such meaningful markup and structure simplifies design. Everyone will benefit from an altogether simpler user experience, one that will be as easy to navigate on any device from a large monitor to a small-screen mobile phone.

Note

CSS Naked Day was supported by one of the cocreators of CSS, Håkon Wium Lie, plus more than 700 designers who shed their virtual clothes and showed the world nothing but their naked content. You can find out more about why so many designers stripped off their styles at http://naked.dustindiaz.com.

Translating meaning into markup: The Markup Is Right

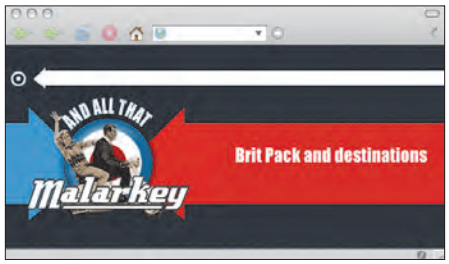
Writing meaningful markup can be simple when you approach it from the content, not the presentation. Rather than starting by asking "What XHTML elements do I need to accomplish this design?" ask yourself "What is this?" and "What does this mean?"

Before you think about the language of XHTML, think about the language you speak and how you would explain the content, and then translate what you have said into markup.

Imagine for a moment you are on a TV quiz show called *The Markup Is Right*. You're a little nervous, but the slick host starts with an easy question to help you feel comfortable.

QUIZ MASTER: "What is this?"

YOU: "It's a top-level heading."



This simply translates to the "most important heading," and in markup, it's an `<h1>`:

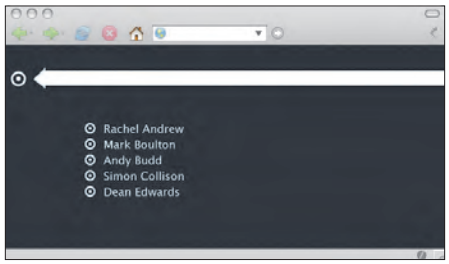
```
<h1>BritPack and destinations</h1>
```

Play on

Luckily, *The Markup Is Right* is not one of those difficult quiz shows with a $100,000,000 top prize; it's more of a teatime quiz where the questions start easy and then don't get a whole lot more difficult. "Let's play on," says the host.

QUIZ MASTER: "What is this?"

YOU: "It's a list of names, in no particular order, and each name is a link."



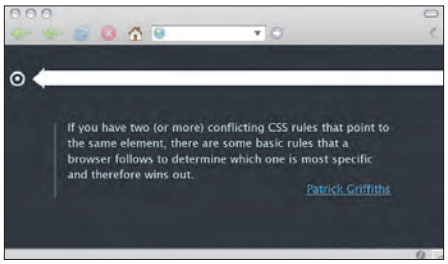
This simply translates to an "unordered list of items (the names) where each link is an anchor," and in markup it looks like this:

```
<ul>
<li><a href="http://www.rachelandrew.co.uk">Rachel Andrew</a></li>
<li><a href="http://www.markboulton.co.uk">Mark Boulton</a></li>
<li><a href="http://www.andybudd.com">Andy Budd</a></li>
<li><a href="http://www.collylogic.com">Simon Collison</a></li>
<li><a href="http://dean.edwards.name">Dean Edwards</a></li>
</ul>
```

You're feeling confident, not even sweating under the lights; the prize is already in sight.

QUIZ MASTER: "You're doing well; the final question for this round is, what is this?"

YOU: "It's a block of text that I am quoting from Patrick Griffiths's HTML Dog Web site."



This translates into markup language as follows:

```
<blockquote>
<p>If you have two (or more) conflicting CSS rules that point to the same
element, there are some basic rules that a browser follows to determine which
one is most specific and therefore wins out.</p>
<p><cite>
<a href="http://www.htmldog.com/">Patrick Griffiths</a>
</cite></p>
</blockquote>
```

You're doing well

At this stage in the game, what the content "looks like" is not important; what matters is meaning. The elements you have chosen have not been for their appearance, and even in basic text browsers, the meaning of the content will be clear.

Picturing the content-out design

After a short break for commercials, you are back in the hot seat, and the smarmy host is aching to get you sweating in this round, the pictures round.

QUIZ MASTER: "I'm going to show you three pictures, and I want you to describe the content you want to convey from each picture and then translate that into markup. Here is your first picture."

NUMBER 1: HORSES

YOU: "It's a picture of several jockeys on horses, each wearing different colored jerseys. The picture has a title, Par for the Horse. Translate that into markup and you have this":

```
<h2>Par for the Horse</h2>
<ul>
<li>Red</li>
<li>Blue</li>
<li>Pink</li>
<li>Green</li>
<ul>
```

You add for a bonus point, "If I wanted a particular horse to link to the horse owner's Web site, I could add an anchor around the color, and because those linked words do not properly describe the contents of the pages I am linking to, I would add a title attribute to those links":

```
<li><a href="http://www.stuffandnonsense.co.uk" title="Andy Clarke's personal
site">Red</a></li>
```


<h2>
Par for the Horse
</h2>

Red

Blue

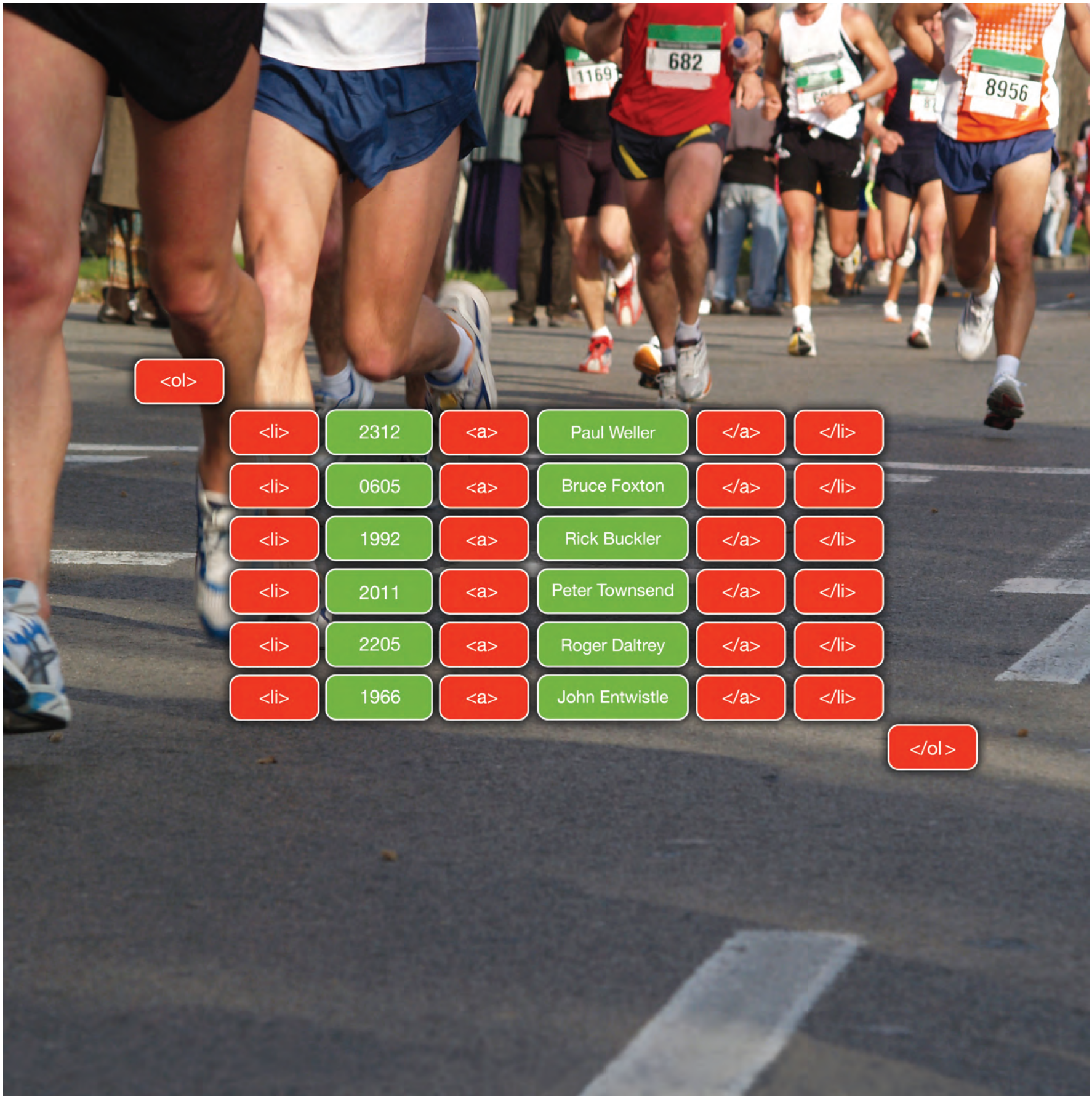
Pink

Green

NUMBER 2: A RACE

YOU: "It's a race and there is an implicit order to the runners. I would also like to list the runners' names and entrance numbers and links to their personal profiles":

```
<ol>
<li>2312 <a href="2312.html">Paul Weller</a></li>
<li>0605 <a href="0605.html">Bruce Foxton</a></li>
<li>1992 <a href="1992.html">Rick Buckler</a></li>
<li>2011 <a href="2011.html">Peter Townsend</a></li>
<li>2205 <a href="2205.html">Roger Daltrey</a></li>
<li>1966 <a href="1966.html">John Entwistle</a></li>
</ol>
```



 2312 <a> Paul Weller
 0605 <a> Bruce Foxton
 1992 <a> Rick Buckler
 2011 <a> Peter Townsend
 2205 <a> Roger Daltrey
 1966 <a> John Entwistle

NUMBER 3: A TAXI QUEUE

YOU: "This one is a little trickier to mark up because there is more detail in the content I want to convey: the taxi number and its driver's name, plus the license plate number and the taxi's position in the rank.

"This information is ideally suited to a table because it is tabular data. I will use table headers to give the content more structure, use a table row for each taxi, and add a caption to the table that describes its contents":

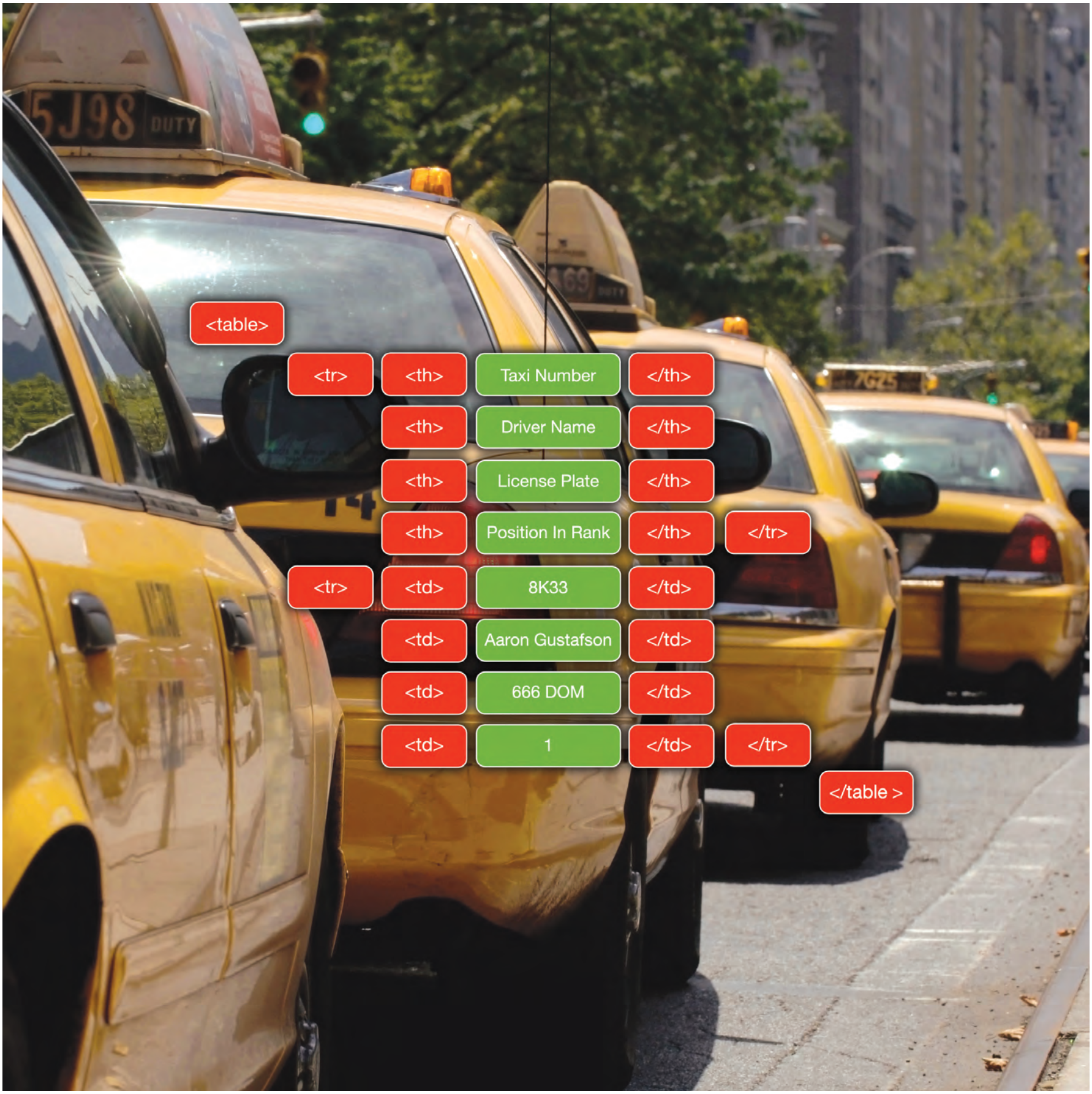
```
<table>
<tr>
<th>Taxi number</th>
<th>Driver name</th>
<th>License plate</th>
<th>Position in rank</th>
</tr>

<tr>
<td>8K33</td>
<td>Aaron Gustafson</td>
<td>666 DOM</td>

<td>1</td>
</tr>

</table>
```

I wish someone gave huge prizes for writing meaningful markup, but the biggest winners will be your visitors. Even if you go home with only a cuddly toy or a toasted sandwich maker, your documents will be smaller, more meaningful, and more accessible if you follow the content-out approach.


<table>
<tr>
<th>
Taxi Number
</th>
<th>
Driver Name
</th>
<th>
License Plate
</th>
<th>
Position In Rank
</th>
</tr>
<tr>
<td>
8K33
</td>
<td>
Aaron Gustafson
</td>
<td>
666 DOM
</td>
<td>
1
</td>
</tr>
</table >

What does the content tell you?

If you are a developer who still works in an environment where designers deliver to you completed design visuals, they can help you choose the most appropriate markup to maintain the meaning of their content in a number of ways.

Designers can add short notes directly to a design visual, perhaps on a separate layer inside a Photoshop or Fireworks file. This can be a powerful way to specify that meaning must be preserved and can also be highly effective when designers and developers work side-by-side and make a simple but effective education or communication tool (**Figure 1.25**).

Meaningful descriptions can include the following:

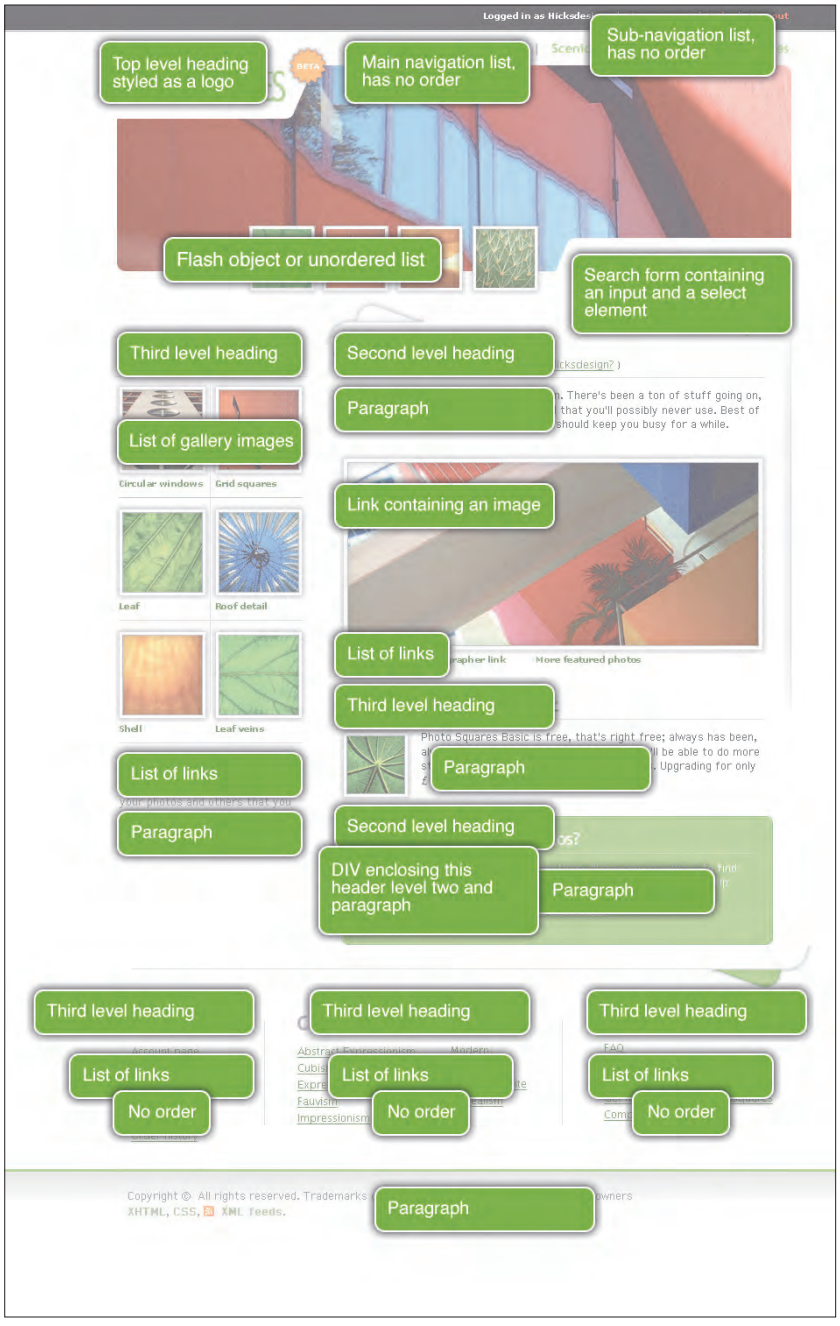
- "This list has no order."
- "An ordered list of top-selling items."
- "A top-level heading."
- "A quotation from a happy customer."

These descriptions contain no presentational information and help clarify the meaning of each design element without a designer ever having to leave the cozy confines of Photoshop.

Moving meaningfully along

By now you have learned that designing for a modern Web requires you to think differently about the ways you write meaningful, semantic markup. Rather than considering the visual layout as your starting point, as so often has been the case until now, you have seen the importance of starting with the "naked" content and working outward, adding appropriately identified divisions until accomplishing your design—all with a minimal amount of markup and none of the presentational hacks that have stalked the Web for so long.

So if you're ready, next you'll learn how to put these ideas into practice by working through a series of short exercises that will help you become more familiar with markup in the Transcendent CSS approach.



1.25 **Clarifying the meaning of each design element**

Example 1

Example 2

Example 3


Example 4


Example 5


Example 6

Example 7


Example 8

Example 9

Marking Up the World

Earlier in this chapter, you learned about the importance of writing meaningful and well-ordered markup as the basis for implementing your designs. You also saw that in the past, visual presentation dictated the elements you chose and the order in which you wrote them. Using the content-out approach, you can free your markup once and for all of presentational thinking. Through some simple examples, you learned that you can, at last, separate meaning and presentation.

Now it's time to put those lessons into practice through a series of fine-art exercises. You'll start by looking at new examples and thinking about the meaning they convey, and then you'll write markup that appropriately describes that meaning. You'll start with lists, the basis of much standards-based code.

All the world's a list; every item must play its part

If you strip away the style sheets from most standards-based pages, you will find an abundance of lists; unordered, ordered, and definition lists have become the stalwarts of meaningful markup.

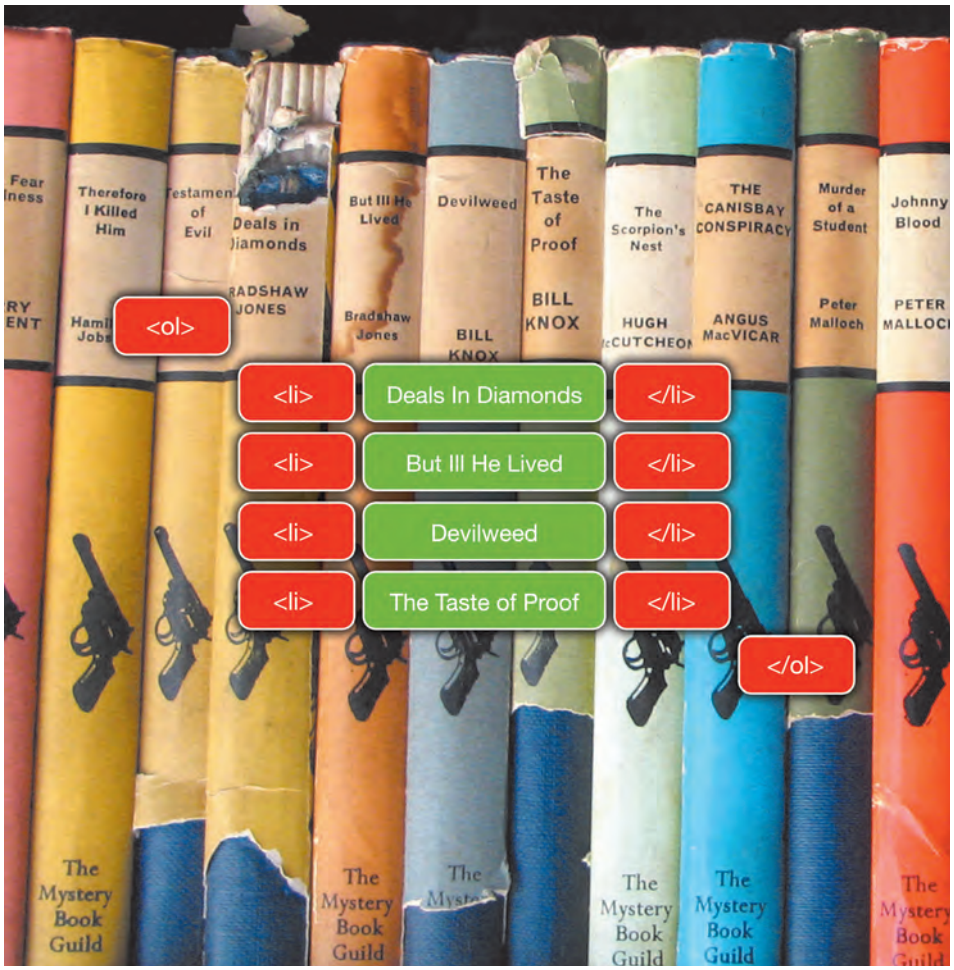
You will find that standards-savvy designers have put lists to many different uses, including everything from simple groups of links to tabbed-style navigation to product listings in e-commerce stores. Some designers have also stretched the semantics of lists and used them for laying out form inputs and their labels.

Lists are all around us, even in the real world. Sure, you have to-do lists and shopping lists, but take the nine examples (**opposite**); in each, a list is appropriate for one or more areas of content. The type of list you choose will depend on the content you want to convey.

Are the books in Example 1 in any order? They may have been stacked on the shelf at random after pulling them straight out of a dusty box. If that were the case, then an unordered list would be the most appropriate. However, it's possible that their owner placed them in the order they were published or, this case, the volume number. If this were true, then the order would dictate that an ordered list would be most appropriate:

```
<ol>
<li>Deals in Diamonds</li>
<li>But Ill He Lived</li>
<li>Devilweed</li>
<li>The Taste of Proof</li>
</ol>
```

Example 1



Now look at Example 5. It would be unlikely that these motorcycles have been parked in order of color or even in order of the year they were made. More likely, their riders parked in the next available space. To list them by model, color, or even the owner name, you would use an unordered list because they were parked in no specific order:

```
<ul>
<li>Green</li>
<li>Gray</li>
<li>Blue</li>
<li>Purple</li>
</ul>
```

Example 5

Example 8

If history is your thing, Example 8 shows three medieval helmets. Perhaps you need to convey their ages and list the helmets in chronological order. However, many times you will need to convey more detailed information about a list of items. In the case of each helmet, you might want to list not only its name but also a short description of its design and how the design affected the wearer's performance in a battle or joust.

Although definition lists were invented strictly to mark up definition terms and their descriptions, many designers have stretched the semantics and used them in instances like this to join "name" and "value" pairs. In this example, you would mark up the name of the helmet using a definition term, `<dt>`, and mark up the description using a definition data element, `<dd>`. This combination of terms and definitions has extended the usefulness of the definition list far beyond that for which it was originally intended:

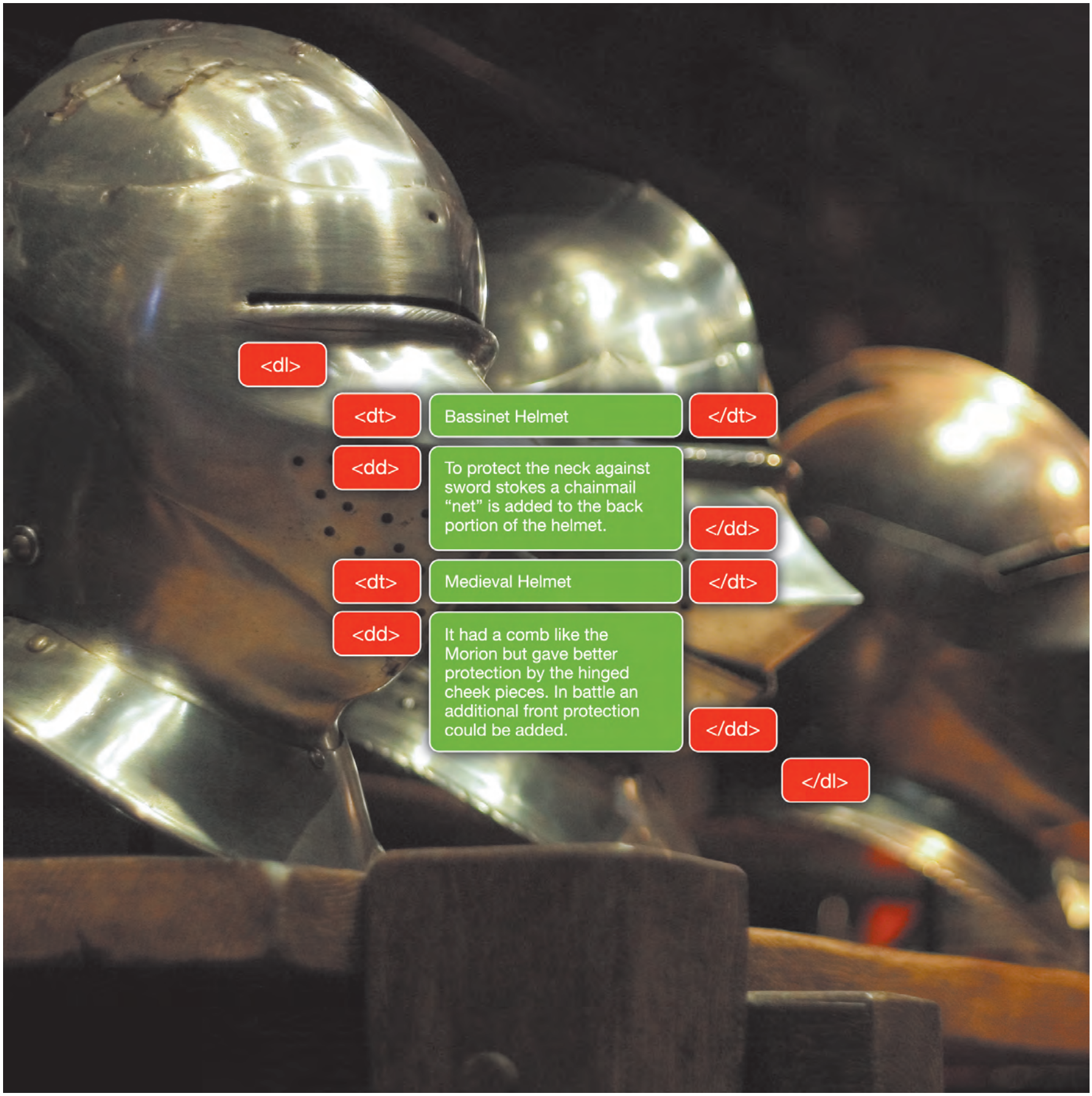
```
<dl>
<dt>Bassinet Helmet</dt>
<dd>To protect the neck against sword strokes a chainmail "net" is added to
the back portion of the helmet.
</dd>

<dt>Medieval Helmet</dt>
<dd>It had a comb like the Morion but gave better protection by the hinged
cheek pieces. In battle an additional front protection could be added.</dd>
</dl>
```

Lists as far as the eye can see

By looking at the remaining examples and considering the content you want to convey with them, you'll find that you'll use lists in every instance. It's your job to hunt them down.

Unfortunately, with only forty XHTML elements at your disposal and only three types of lists, sometimes you will need to combine lists with other elements to give them more precise meaning.


<dl>
<dt>
Bassinet Helmet
</dt>
<dd>
To protect the neck against sword stokes a chainmail "net" is added to the back portion of the helmet.
</dd>
<dt>
Medieval Helmet
</dt>
<dd>
It had a comb like the Morion but gave better protection by the hinged cheek pieces. In battle an additional front protection could be added.
</dd>
</dl>

Note

What's an XHTML compound, you ask? To learn more about XHTML and XHTML compounds, you can download the full presentation of "The Elements of Meaningful XHTML" at http://tantek.com/presentations/2005/09/elements-of-xhtml/.

"If I have two beans and then add two more beans, what do I have?"

Often you can use lists in combination with other structural elements to create new XHTML *compounds*. XHTML compounds provide meaning that is more specific than is possible with just a single element.

I'll use the example of a conversation between two of my favorite comedy characters, Captain Blackadder and Private Baldrick from *Blackadder Goes Forth*. What elements should you use to mark up this conversation, given that no `<conversation>` element exists in XHTML?

Headings and paragraphs are not precise enough to say *conversation*, and although many designers might be tempted to use a definition list, a conversion is not strictly *terms* or *definitions*. What is needed is a combination of meaningful elements that together form a compound to describe a conversation. Building precise XHTML compounds are simple when you design from the content out.

A plan so cunning you could brush your teeth with it

What are conversations, and what happens in them? No matter what is said or who says what, in conversations people say words, and they do so over a period of time.

In XHTML you already have the `<blockquote>` element to describe what a person says:

```
<blockquote>
<p>The British Empire at present covers a quarter of the globe, while the
German Empire consists of a small sausage factory in Tanganyika. I hardly
think that we can be entirely absolved of blame on the imperialistic front.
</p>
</blockquote>
```

You also have the `<cite>` element for the name of the person who is being quoted:

```
<cite>Captain Blackadder</cite>
```

What is missing now is an element that can group who is speaking, and what they're saying, in the order that took place during the conversation.

From the speakers and quotations at the center, you can work outward, wrapping that content inside another element that was designed to add order; cunningly, it was named an *ordered* list:

```
<ol>
<li>
<cite>Blackadder</cite>
<blockquote>
<p>If I have two beans and then I add two more beans, what do I have?</p>
</blockquote>
</li>

<li>
<cite>Baldrick</cite>
<blockquote>
<p>Some beans</p>
</blockquote>
</li>

<li>
<cite>Blackadder</cite>
<blockquote>
<p>Yes and no. Let's try again shall we? I have two beans, then I add two more
beans. What does that make?</p>
</blockquote>
</li>

<li>
<cite>Baldrick</cite>
<blockquote>
<p>A very small casserole.</p>
</blockquote>
</li>

<li>
<cite>Blackadder</cite>
<blockquote>
<p>Yes. To you Baldrick, the Renaissance was just something that happened to
other people, wasn't it?</p>
</blockquote>
</li>
</ol>
```

Combining elements that have rich, semantic meaning to form compounds is an ideal way to extend the meaning of your content.

Note

The <address> element is possibly one of the most badly named elements in XHTML. It was not designed for use with physical addresses but with contact information for a particular page, such as the e-mail address for the author of the content.

Send me an hCard from San Francisco

Take a look at this photograph of San Francisco, one of my favorite cites in the world. Like San Francisco itself, a lot is going on here; for instance, you can see a lot of streets with a lot of buildings that contain a lot of businesses.

How do you think you would mark up a city like this? "With more markup than there is room for in this book" would be one answer, but not the one I want. A more correct answer would be "It depends." Specifically, it depends on the information you are aiming to convey.

Like all towns and cities great and small, San Francisco has streets, and those streets have numbered buildings. This implies that if you were aiming to list the buildings in any one street, then an ordered list would be the most appropriate element:

```
<ol>
<li>665 3rd Street, San Francisco, 94107, California</li>
</ol>
```

What if you are interested in marking up the address of a particular building? Your first thought might be to grab hold of the nearest <address> element. Sadly, you would be wrong.

Despite the unfortunate name of the <address> element, no element in XHTML was designed for use with a physical address or location. And no elements exist that can adequately describe a street, postal code, or state.

Earlier you were introduced to the concept of microformats to add more precise meaning by using special class attributes on some elements. Here too you can work from the content out and use microformats, in particular the *hCard* microformat, to create meaning in your page.

You can enclose each line of your address in a element and give each line a precise attribute to reflect the contents of that line:

```
<span class="street-address">665 3rd Street</span>
<span class="locality">San Francisco</span>
<span class="postal-code">94107</span>
<span class="region">California</span>
```

In place of an <address> element, you should provide a context for these address parts by enclosing them all in another element:

```
<span class="adr">
<span class="street-address">665 3rd Street</span>
<span class="locality">San Francisco</span>
<span class="postal-code">94107</span>
<span class="region">California</span>
</span>
```

Finally, you should add a class attribute of vcard to the list item to ensure both real people and software applications can read the complete compound, one of the key benefits of using microformats.

Learning to keep your eyes wide open

Looking around you and noting the implicit meaning of what you see can become an interesting, if a little geeky, way to pass the time on journeys to and from your studio. Wherever you go, you can extract meaning from the following:

- Advertising hoardings and billboards
- Signage
- Shop window displays

In fact, you can extract meaning from almost anything in the world. If you, like me, don't get out as often as you should, look for semantics in the pages of newspapers or magazines or on packaging. You'll be surprised at how much nutritional, semantic information appears on your box of morning cereal.

You should start seeing markup appearing before your eyes, everywhere you look. But if you start hearing voices, I suggest you contact your doctor immediately.

Working from the "contents"

In fact, if you are similar at all to me, when you start thinking about the semantic meaning of the objects around you, you will notice yourself wondering how to mark up almost everything you touch.

Flicking through the pages of magazines and newspapers, you will begin to wonder not only what markup would be most appropriate to the content, but also how you might accomplish a design element with CSS.

Given the example of a Contents page from a gardening magazine (**Figure 1.26**), let's work through how you might arrive at the meaningful markup to represent it.

1.26 **Taking on a magazine layout**

With an initial glance, you might first think about dividing this page into two columns, using two divisions: one for the written content on the left and one on the right to hold the images. But presentational thinking often results in presentational markup, and working outward from the content will help you achieve minimal, meaningful markup.

Scribble notes

You can use different techniques to help you visualize markup in cuttings you make from newspapers or magazines; one of my favorite methods is to write notes either on the cutting or on tracing paper taped over the cutting in my scrapbook (**Figure 1.27**). You can scribble notes about the meaning of the content and your thoughts about the most appropriate elements.

1.27 **Making notes**

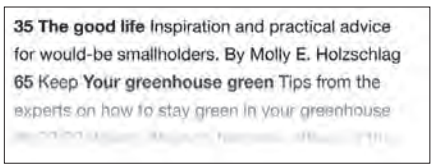
Visualize the magazine markup

Looking at this example, a number of elements spring immediately to mind:

- A top-level heading, <h1>, for the word *Contents* because it is the most important heading on the page

- Second-level headings, <h2>, for the words *Gardens & Houses* and *Features*

- All contained in a division: a lower-level heading, <h3>; a list of page numbers and titles that are featured on the cover, <ul>; and a thumbnail picture of the cover of the magazine, <img>

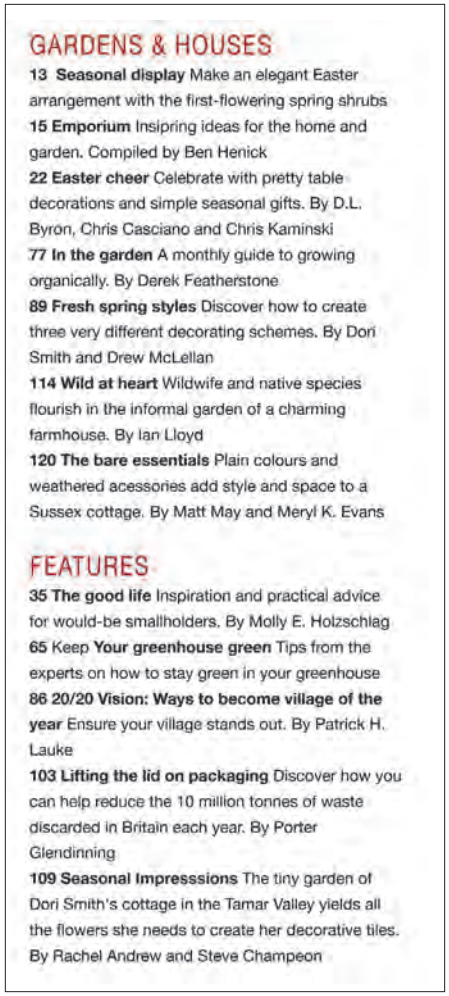


But what about the two lists of articles and the images on the right? These require a little more abstract thinking. But you might be surprised at just how simple the solutions to problems like this can be when you start at the essence of the content.

Each contains the "title" of an internal page and a short description of what the reader might find on that page if she turns to it. Headings—in this case, third-level headings, <h3>—can form the titles of the pages, and simple paragraphs, <p>, make up the descriptions:

```
<h3>Seasonal Display</h3>

<p>Make an elegant Easter arrangement with the first-flowering spring shrubs</p>
```



GARDENS & HOUSES

13 Seasonal display Make an elegant Easter arrangement with the first-flowering spring shrubs

15 Emporium Inspiring ideas for the home and garden. Compiled by Ben Henick

22 Easter cheer Celebrate with pretty table decorations and simple seasonal gifts. By D.L. Byron, Chris Casciano and Chris Kaminski

77 In the garden A monthly guide to growing organically. By Derek Featherstone

89 Fresh spring styles Discover how to create three very different decorating schemes. By Dori Smith and Drew McLellan

114 Wild at heart Wildwife and native species flourish in the informal garden of a charming farmhouse. By Ian Lloyd

120 The bare essentials Plain colours and weathered acessories add style and space to a Sussex cottage. By Matt May and Meryl K. Evans

FEATURES

35 The good life Inspiration and practical advice for would-be smallholders. By Molly E. Holzschlag

65 Keep Your greenhouse green Tips from the experts on how to stay green in your greenhouse

86 20/20 Vision: Ways to become village of the year Ensure your village stands out. By Patrick H. Lauke

103 Lifting the lid on packaging Discover how you can help reduce the 10 million tonnes of waste discarded in Britain each year. By Porter Glendinning

109 Seasonal Impresssions The tiny garden of Dori Smith's cottage in the Tamar Valley yields all the flowers she needs to create her decorative tiles. By Rachel Andrew and Steve Champeon

1.28 Ordering the list

Order, please

Because the pages are listed in the order they appear in the magazine, an ordered list, `<ol>`, will give extra meaning to the lists (**Figure 1.28**):

```
<ol>
<li>
<h3>The Good Life</h3>
<p>Inspiration and practical advice for would-be smallholders. By Molly E.
Holzschlag.</p>
</li>

<li>
<h3>20/20 Vision: Ways to become village of the year</h3>
<p>Ensure your village stands out. By Patrick H. Lauke</p>
[etc.]
</ol>
```

Add in links

You will also need to add links to other pages inside the headings for each article; after all, you can't have a page without at least one link, can you?

```
<li>
<h3><a href="81.html">20/20 Vision: Ways to become village of the year</a></h3>
<p>Ensure your village stands out. By <a href="lauke.html">Patrick H. Lauke</a>
</p>
</li>
```

That is essentially the entire structural markup you will need to convey the meaning of this page's content.

At this point you might be wondering, but what about the images? Where are the `<img>` elements? Where will they appear in the flow of the document? Remember when I said the images would require a little more abstract thinking? Looking again at the images, you will see that their *function* is to *highlight* particular articles by illustrating them with an image and giving the reader the page number of that article. In essence, their function is no different from any of the other article links you have so meaningfully placed inside your ordered lists (**Figure 1.29**).

Place emphasis

Your next task is to identify these important articles individually and give them all additional semantic *emphasis* in your markup.

You can start by giving the list item for these special articles an individual identity, logically the page number on which the articles appear. Because XHTML does not allow an `id` attribute to begin with a numeral, you can prefix your `id` attribute with a letter, in this case *p* (for *page*):

```
<li id="p89">
<h3><a href="89.html">Fresh spring styles</a></h3>
<p>Discover how to create three very different decorating schemes. By
<a href="smith.html">Dori Smith</a> and <a href="mclellan.html">Drew McLellan
</a></p>
</li>
```

All four of these articles should be emphasized in some way as being featured articles; luckily, the emphasis element, `<em>`, is waiting and willing to help you.

You will wrap the emphasis around the name of each article and its link to the article page:

```
<li id="p89">
<h3><em><a href="89.html">Fresh spring styles</a></em></h3>
<p>Discover how to create three very different decorating schemes. By
<a href="smith.html">Dori Smith</a> and <a href="mclellan.html">Drew McLellan
</a></p>
</li>
```

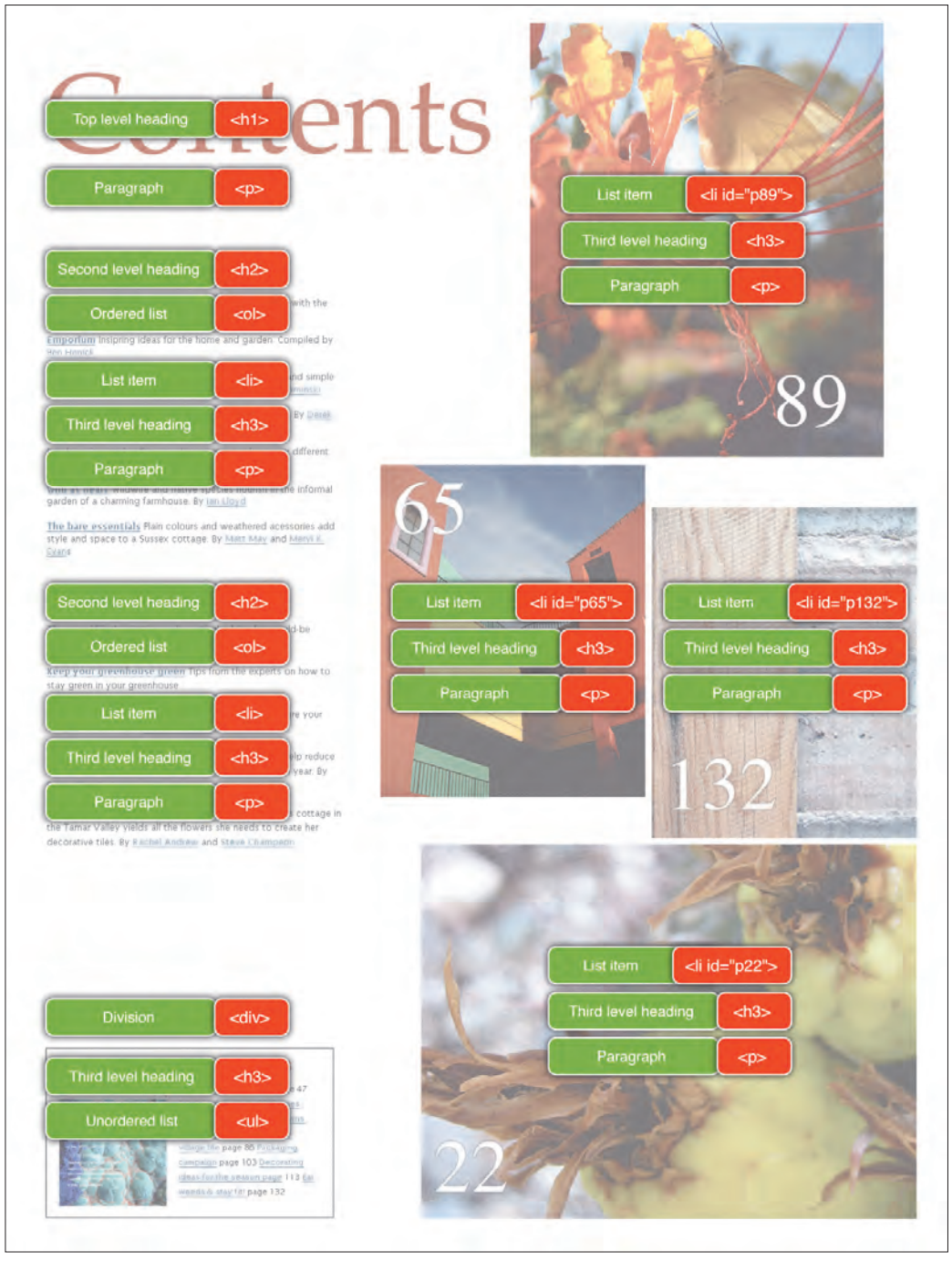
You will use these `id` attributes and emphasis to style these featured articles differently than you style all the others.

1.29 **Highlighting particular articles**

Complete the markup

Your completed markup has created a meaningful, well-ordered flow of "contents" ready to be styled with CSS. Take a look at the completed markup, and compare it to the markup guide to see the relationships between your elements and how the completed page will look when styled with CSS (**Figure 1.30 and Figure 1.31, next page**).

1.30 **Creating a well-ordered flow of contents**



1.31 **Marking up the content**

Time to Process What You Have Learned

Having learned about how to look for meaning in unusual places, how to give your markup more precise meaning by combining elements into "compounds" and using microformats, and how to separate the natural order of your content from your visual goals, it's time to start processing all this new information.

In Part 2, "Process," you will take a fresh look at the design and development process. You will learn some exciting new ways to start your designs, learn how to use wireframes more effectively, and go step-by-step through the process of turning a design into a prototype using meaningful markup and CSS.