

A Brain-Friendly Guide

2nd  
Edition  
Updated for HTML5

# Head First HTML and CSS

Launch your  
web career in  
one chapter



A learner's guide  
to creating  
standards-based  
web pages



Watch out for  
common HTML &  
CSS traps and pitfalls

## Free Sampler

Bend your mind  
around 100 puzzles  
& exercises



Learn why everything  
your friends know about  
style is probably wrong

Avoid  
embarrassing  
validation mistakes



O'REILLY®

Elisabeth Robson & Eric Freeman

## **Head First HTML and CSS**

by Elisabeth Robson and Eric Freeman

Copyright © 2012 Elisabeth Robson and Eric Freeman. All rights reserved.

Printed in Canada.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly Media books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (<http://my.safaribooksonline.com>). For more information, contact our corporate/institutional sales department: (800) 998-9938 or [corporate@oreilly.com](mailto:corporate@oreilly.com).

**Series Creators:** Kathy Sierra, Bert Bates

**Editor:** Brett McLaughlin (first edition), Mike Hendrickson (second edition)

**Cover Designer:** Karen Montgomery

**HTML Wranglers:** Elisabeth Robson, Eric Freeman

**Production Editor:** Kristen Borg

**Indexer:** Ron Strauss

**Proofreader:** Rachel Monaghan

**Page Viewer:** Oliver



**Printing History:**

December 2005: First Edition.

September 2012: Second Edition.

Nutshell Handbook, the Nutshell Handbook logo, and the O'Reilly logo are registered trademarks of O'Reilly Media, Inc. The *Head First* series designations, *Head First HTML and CSS*, and related trade dress are trademarks of O'Reilly Media, Inc.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly Media, Inc., was aware of a trademark claim, the designations have been printed in caps or initial caps.

While every precaution has been taken in the preparation of this book, the publisher and the authors assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

In other words, if you use anything in *Head First HTML and CSS* to, say, run a nuclear power plant, you're on your own. We do, however, encourage you to visit the Head First Lounge.

No elements or properties were harmed in the making of this book.

Thanks to Clemens Orth for the use of his photo, "applestore.jpg", which appears in Chapter 5.

ISBN: 978-0-596-15990-0

[TI]

# Table of Contents (summary)

	Intro	xxv
1	The Language of the Web: <i>getting to know html</i>	1
2	Meet the “HT” in HTML: <i>going further, with hypertext</i>	43
3	Web Page Construction: <i>building blocks</i>	77
4	A Trip to Webville: <i>getting connected</i>	123
5	Meeting the Media: <i>adding images to your pages</i>	163
6	Serious HTML: <i>standards and all that jazz</i>	219
7	Adding a Little Style: <i>getting started with CSS</i>	255
8	Expanding your Vocabulary: <i>styling with fonts and colors</i>	311
9	Getting Intimate with Elements: <i>the box model</i>	361
10	Advanced Web Construction: <i>divs and spans</i>	413
11	Arranging Elements: <i>layout and positioning</i>	471
12	Modern HTML: <i>html5 markup</i>	545
13	Getting Tabular: <i>tables and more lists</i>	601
14	Getting Interactive: <i>html forms</i>	645
	Appendix: The Top Ten Topics (We Didn’t Cover): <i>leftovers</i>	697

# Table of Contents (the real thing)

## Intro

**Your brain on HTML and CSS.** Here you are trying to *learn* something, while here your *brain* is doing you a favor by making sure the learning doesn’t *stick*. Your brain’s thinking, “Better leave room for more important things, like which wild animals to avoid and whether naked snowboarding is a bad idea.” So how *do* you trick your brain into thinking that your life depends on knowing HTML and CSS?

Who is this book for?	xxvi
Metacognition	xxix
Here’s what WE did	xxx
Bend your brain into submission	xxxi
Tech reviewers (first edition)	xxxiv
Acknowledgments (first edition)	xxxv
Tech reviewers (second edition)	xxxvi
Acknowledgments (second edition)	xxxvi

# getting to know html

## 1

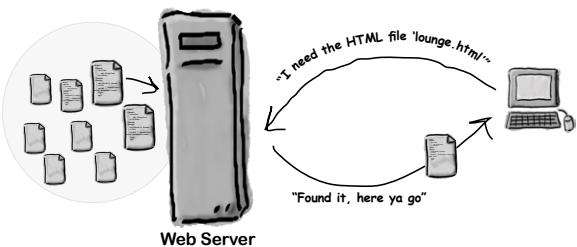
### The Language of the Web

The only thing that is standing between you and getting yourself on the Web is learning to speak the lingo:

HyperText Markup Language, or HTML for short. So, get ready for some language lessons. After this chapter, not only are you going to understand some basic **elements** of HTML, but you'll also be able to speak HTML with a little **style**. Heck, by the end of this book, you'll be talking HTML like you grew up in Webville.



The web killed the radio star	2
What does the web server do?	3
What you write (the HTML)	4
What the browser creates	5
Your big break at Starbuzz Coffee	9
Creating the Starbuzz web page	11
Creating an HTML file (Mac)	12
Creating an HTML file (Windows)	14
Meanwhile, back at Starbuzz Coffee...	17
Saving your work	18
Opening your web page in a browser	19
Take your page for a test drive	20
Are we there yet?	23
Another test drive	24
Tags dissected	25
Meet the style element	29
Giving Starbuzz some style...	30
Cruisin' with style...	31
Exercise Solutions	38



## going further with hypertext

# 2

## Meeting the “HT” in HTML

**Did someone say “hypertext?” What’s that?** Oh, only the entire basis of the Web. In Chapter 1 we kicked the tires of HTML and found it to be a nice markup language (the “ML” in HTML) for describing the structure of web pages. Now we’re going to check out the “HT” in HTML, hypertext, which will let us break free of a single page and link to other pages. Along the way we’re going to meet a powerful new element, the `<a>` element, and learn how being “relative” is a groovy thing. So, fasten your seat belts—you’re about to learn some hypertext.



Head First Lounge, <i>new and improved</i>	44
Creating the new lounge	46
What did we do?	48
Understanding attributes	51
Getting organized	56
Organizing the lounge...	57
Technical difficulties	58
Planning your paths...	60
Fixing those broken images...	66
Exercise Solutions	73



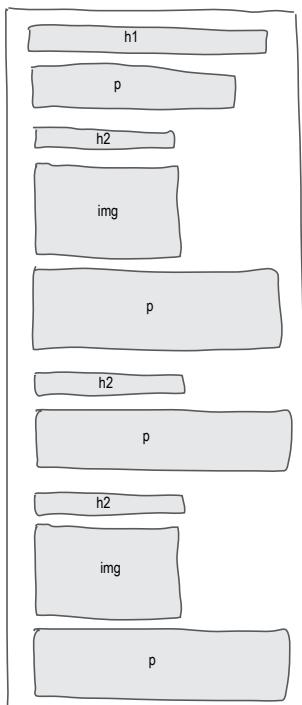
# building blocks

# 3

## Web Page Construction

### I was told I'd actually be creating web pages in this book?

You've certainly learned a lot already: tags, elements, links, paths...but it's all for nothing if you don't create some killer web pages with that knowledge. In this chapter we're going to ramp up construction: you're going to take a web page from conception to blueprint, pour the foundation, build it, and even put on some finishing touches. All you need is your hard hat and your toolbelt, as we'll be adding some new tools and giving you some insider knowledge that would make Tim "The Toolman" Taylor proud.



From journal to website, at 12 mph	79
The rough design sketch	80
From a sketch to an outline	81
From the outline to a web page	82
Test-driving Tony's web page	84
Adding some new elements	85
Meet the <q> element	86
Looooong quotes	90
Adding a blockquote	91
The real truth behind the <q> and <blockquote> mystery	94
Meanwhile, back at Tony's site...	100
Of course, you could use the <p> element to make a list...	101
Constructing HTML lists in two easy steps	102
Taking a test drive through the cities	104
Putting one element inside another is called "nesting"	107
To understand the nesting relationships, draw a picture	108
Using nesting to make sure your tags match	109
Exercise Solutions	117

# getting connected

# 4

## A Trip to Webville

**Web pages are a dish best served on the Internet.** So far you've only created HTML pages that live on your own computer. You've also only linked to pages that are on your own computer. We're about to change all that. In this chapter we'll encourage you to get those web pages on the Internet where all your friends, fans, and customers can actually see them. We'll also reveal the mysteries of linking to other pages by cracking the code of the h, t, t, p, :, /, /, w, w, w. So, gather your belongings; our next stop is Webville.

Getting Starbuzz (or yourself) onto the Web	124
Finding a hosting company	125
How can you get a domain name?	126
Moving in	128
Getting your files to the root folder	129
As much FTP as you can possibly fit in two pages	130
Back to business...	133
Mainstreet, USA	134
What is HTTP?	135
What's an absolute path?	136
How default pages work	139
Earl needs a little help with his URLs	140
How do we link to other websites?	142
Linking to Caffeine Buzz	143
And now for the test drive...	144
Web page fit and finish	147
The title test drive...	148
Linking into a page	149
Using the id attribute to create a destination for <a>	150
How to link to elements with ids	151
Linking to a new window	155
Opening a new window using target	156
Exercise Solutions	160

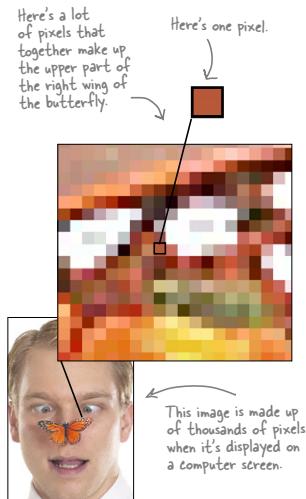


## adding images to your pages

# 5

## Meeting the Media

Smile and say “cheese.” Actually, smile and say “gif,” “jpg,” or “png”—these are going to be your choices when “developing pictures” for the Web. In this chapter you’re going to learn all about adding your first media type to your pages: images. Got some digital photos you need to get online? No problem. Got a logo you need to get on your page? Got it covered. But before we get into all that, don’t you still need to be formally introduced to the `<img>` element? So sorry, we weren’t being rude; we just never saw the “right opening.” To make up for it, here’s an entire chapter devoted to `<img>`. By the end of the chapter you’re going to know all the ins and outs of how to use the `<img>` element and its attributes. You’re also going to see exactly how this little element causes the browser to do extra work to retrieve and display your images.



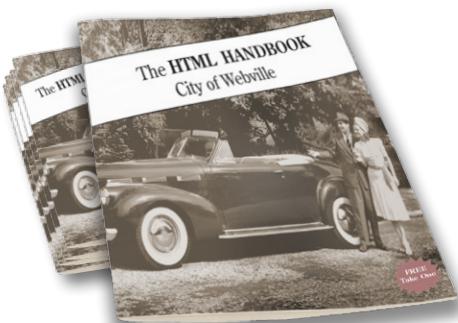
How the browser works with images	164
How images work	167
<code>&lt;img&gt;</code> : it's not just relative links anymore	171
Always provide an alternative	173
Sizing up your images	174
Creating the ultimate fan site: myPod	175
Whoa! The image is way too large	178
Open the image	182
Resizing the image	183
Fixing up the myPod HTML	188
More photos for myPod	190
Turning the thumbnails into links	196
Create individual pages for the photos	197
So, how do I make links out of images?	198
What format should we use?	203
To be transparent, or not to be transparent? That is the question...	204
Wait, what is the color of the web page background?	206
Check out the logo with a matte	207
Add the logo to the myPod web page	208
Exercise Solutions	213

## standards and all that jazz

# 6

### Serious HTML

**What else is there to know about HTML?** You're well on your way to mastering HTML. In fact, isn't it about time we move on to CSS and learn how to make all this bland markup look fabulous? Before we do, we need to make sure your HTML is really ready for the big leagues. Don't get us wrong, you've been writing first-class HTML all along, but there are just a few extra things you need to do to make it "industry standard" HTML. It's also time you think about making sure you're using the latest and greatest HTML standard, otherwise known as HTML5. By doing so, you'll ensure that your pages play well with the latest i-Device, and that they'll display more uniformly across all browsers (at least the ones you'd care about). You'll also have pages that load faster, pages that are guaranteed to play well with CSS, and pages that are ready to move into the future as the standards grow. Get ready, this is the chapter where you move from web tinkerer to web professional.



A Brief History of HTML	222
The new, and improved, HTML5 doctype	227
HTML, the new "living standard"	228
Adding the document type definition	229
The doctype test drive	230
Meet the W3C validator	233
Validating the Head First Lounge	234
Houston, we have a problem...	235
Fixing that error	236
We're almost there...	237
Adding a <meta> tag to specify the character encoding	239
Making the validator (and more than a few browsers) happy with a <meta> tag...	240
Third time's the charm?	241
Calling all HTML professionals, grab the handbook...	244
Exercise Solutions	251

# getting started with CSS

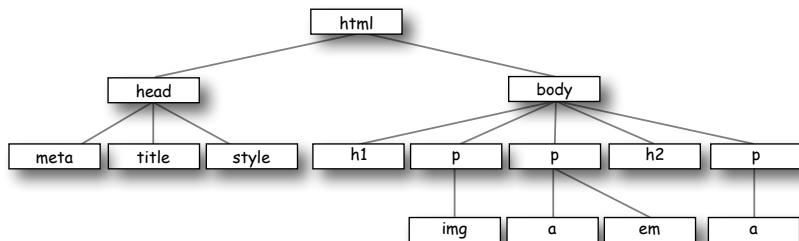
# 7

## Adding a Little Style

I was told there'd be CSS in this book. So far you've been concentrating on learning HTML to create the structure of your web pages. But as you can see, the browser's idea of style leaves a lot to be desired. Sure, we could call the fashion police, but we don't need to. With CSS, you're going to completely control the presentation of your pages, often without even changing your HTML. Could it really be so easy? Well, you *are* going to have to learn a new language; after all, Webville is a bilingual town. After reading this chapter's guide to learning the language of CSS, you're going to be able to stand on *either* side of Main Street and hold a conversation.



You're not in Kansas anymore	256
Overheard on Webville's "Trading Spaces"	258
Using CSS with HTML	259
Getting CSS into your HTML	261
Adding style to the lounge	262
Let's put a line under the welcome message too	265
So, how do selectors really work?	267
Seeing selectors visually	270
Getting the Lounge style into the elixirs and directions pages	273
It's time to talk about your inheritance...	281
Overriding inheritance	284
Adding an element to the greentea class	287
Creating a class selector	288
Taking classes further...	290
The world's smallest and fastest guide to how styles are applied	292
Exercise Solutions	303



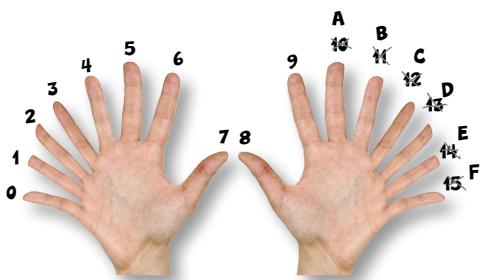
## styling with fonts and colors

# 8

## Expanding Your Vocabulary

Your CSS language lessons are coming along nicely. You already have the basics of CSS down, and you know how to create CSS rules to select and specify the style of an element. Now it's time to build your vocabulary, and that means picking up some new properties and learning what they can do for you. In this chapter we're going to work through some of the most common properties that affect the display of text. To do that, you'll need to learn a few things about fonts and color. You're going to see you don't have to be stuck with the fonts everyone else uses, or the clunky sizes and styles the browser uses as the defaults for paragraphs and headings. You're also going to see there is a lot more to color than meets the eye.

Text and fonts from 30,000 feet	312
What is a font family anyway?	314
Specifying font families using CSS	317
Dusting off Tony's journal	318
How do I deal with everyone having different fonts?	321
How Web Fonts work	323
How to add a Web Font to your page...	325
Adjusting font sizes	328
So, how should I specify my font sizes?	330
Let's make these changes to the font sizes in Tony's web page	332
Changing a font's weight	335
Adding style to your fonts	337
Styling Tony's quotes with a little italic	338
How do web colors work?	340
How do I specify web colors? Let me count the ways...	343
The two-minute guide to hex codes	346
How to find web colors	348
Back to Tony's page...	351
Everything you ever wanted to know about text-decorations	353
Removing the underline...	354
Exercise Solutions	357



## 9

## the box model

**Getting Intimate with Elements**

**To do advanced web construction, you really need to know your building materials.** In this chapter we're going to take a close look at our building materials: the HTML elements. We're going to put block and inline elements right under the microscope and see what they're made of. You'll see how you can control just about every aspect of how an element is constructed with CSS. But we don't stop there—you'll also see how you can give elements unique identities. And, if that weren't enough, you're going to learn when and why you might want to use multiple stylesheets. So, turn the page and start getting intimate with elements.

The lounge gets an upgrade	362
Starting with a few simple upgrades	364
Checking out the new line height	366
Getting ready for some major renovations	367
A closer look at the box model	368
What you can do to boxes	370
Creating the guarantee style	375
A test drive of the paragraph border	376
Padding, border, and margins for the guarantee	377
Adding a background image	380
Fixing the background image	383
How do you add padding only on the left?	384
How do you increase the margin just on the right?	385
A two-minute guide to borders	386
Border fit and finish	389
Using an id in the lounge	396
Using multiple stylesheets	399
Stylesheets—they're not just for desktop browsers anymore...	400
Add media queries right into your CSS	401
Exercise Solutions	407



# 10

## divs and spans

### Advanced Web Construction

**It's time to get ready for heavy construction.** In this chapter we're going to roll out two new HTML elements: <div> and <span>. These are no simple "two by fours"; these are full-blown steel beams. With <div> and <span>, you're going to build some serious supporting structures, and once you've got those structures in place, you're going to be able to style them all in new and powerful ways. Now, we couldn't help but notice that your CSS toolbelt is really starting to fill up, so it's time to show you a few shortcuts that will make specifying all these properties a lot easier. And we've also got some special guests in this chapter, the *pseudo-classes*, which are going to allow you to create some very interesting selectors. (If you're thinking that "pseudo-classes" would make a great name for your next band, too late; we beat you to it.)



A close look at the elixirs HTML	415
Let's explore how we can divide a page into logical sections	417
Adding a border	424
Adding some real style to the elixirs section	425
Working on the elixir width	426
Adding the basic styles to the elixirs	431
What we need is a way to select descendants	437
Changing the color of the elixir headings	439
Fixing the line height	440
It's time to take a little shortcut	442
Adding <span>s in three easy steps	448
The <a> element and its multiple personalities	452
How can you style elements based on their state?	453
Putting those pseudo-classes to work	455
Isn't it about time we talk about the "cascade"?	457
The cascade	459
Welcome to the "What's my specificity?" game	460
Putting it all together	461
Exercise Solutions	467

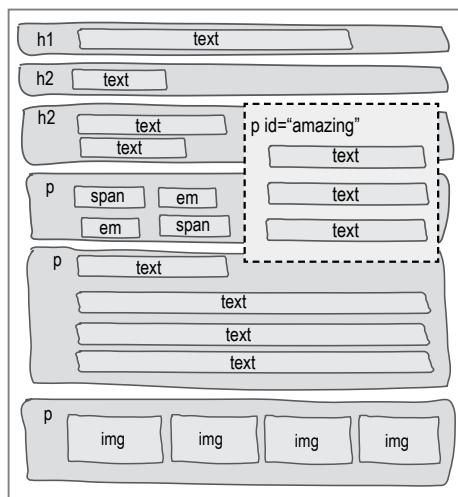
## 11

## layout and positioning

## Arranging Elements



**It's time to teach your HTML elements new tricks.** We're not going to let those HTML elements just sit there anymore—it's about time they get up and help us create some pages with real *layouts*. How? Well, you've got a good feel for the `<div>` and `<span>` structural elements and you know all about how the box model works, right? So, now it's time to use all that knowledge to craft some real designs. No, we're not just talking about more background and font colors—we're talking about full-blown professional designs using multicolumn layouts. This is the chapter where everything you've learned comes together.



Did you do the Super Brain Power?	472
Use the Flow, Luke	473
What about inline elements?	475
How it all works together	476
How to float an element	479
The new Starbuzz	483
Move the sidebar just below the header	488
Fixing the two-column problem	491
Setting the margin on the main section	492
Solving the overlap problem	495
Righty tighty, lefty loosey	498
Liquid and frozen designs	501
How absolute positioning works	504
Changing the Starbuzz CSS	507
How CSS table display works	511
Adding HTML structure for the table display	513
What's the problem with the spacing?	517
Problems with the header	524
Fixing the header images with float	525
Positioning the award	528
How does fixed positioning work?	531
Using a negative left property value	533
Exercise Solutions	539

## html5 markup

### Modern HTML

# 12

**So, we're sure you've heard the hype around HTML5.** And, given how far along you are in this book, you're probably wondering if you made the right purchase. Now, one thing to be clear about, up front, is that everything you've learned in this book has been HTML, and more specifically has met the HTML5 standard. But there are some new aspects of HTML markup that were added with the HTML5 standard that we haven't covered yet, and that's what we're going to do in this chapter. Most of these additions are evolutionary, and you're going to find you are quite comfortable with them given all the hard work you've already done in this book. There's some revolutionary stuff too (like video), and we'll talk about that in this chapter as well. So, let's dive in and take a look at these new additions!



Rethinking HTML structure	546
Update your Starbuzz HTML	551
How to update your CSS for the new elements	554
Setting up the CSS for the blog page	563
We still need to add a date to the blog...	565
Adding the <time> element to your blog	566
How to add more <header> elements	568
So, what's wrong with the header anyway?	570
A final test drive for the headers	571
Completing the navigation	574
Who needs GPS? Giving the navigation a test drive	575
Ta-da! Look at that navigation!	577
Creating the new blog entry	580
Lights, camera, action...	581
How does the <video> element work?	583
Closely inspecting the video attributes...	584
What you need to know about video formats	586
The video format contenders	587
How to juggle all those formats...	589
How to be even more specific with your video formats	590
Exercise Solutions	597

# 13

## tables and more lists

### Getting Tabular

If it walks like a table and talks like a table... There comes a time in life when we have to deal with the dreaded *tabular data*. Whether you need to create a page representing your company's inventory over the last year or a catalog of your vinylmation collection (don't worry, we won't tell), you know you need to do it in HTML, but how? Well, have we got a deal for you: order now, and in a single chapter we'll reveal the secrets that will allow you to put your very own data right inside HTML tables. But there's more: with every order we'll throw in our exclusive guide to styling HTML tables. And, if you act now, as a special bonus, we'll throw in our guide to styling HTML lists. Don't hesitate; call now!

How do you make tables with HTML?	603
Creating a table with HTML	604
What the browser creates	605
Tables dissected	606
Adding a caption	609
Before we start styling, let's get the table into Tony's page	611
Getting those borders to collapse	616
How about some color?	618
Tony made an interesting discovery	620
Another look at Tony's table	621
How to tell cells to span more than one row	622
Test drive the table	624
Trouble in paradise?	625
Overriding the CSS for the nested table headings	629
Giving Tony's site the final polish	630
What if you want a custom marker?	632
Exercise Solutions	636



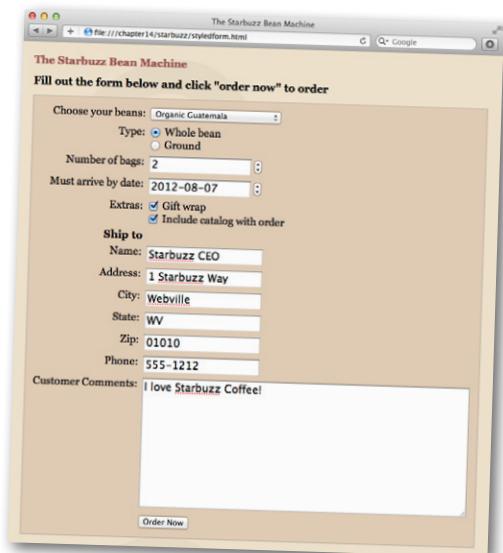
City	Date	Temperature	Altitude	Population	Diner Rating					
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5					
Magic City, ID	June 25th	74	5,312 ft	50	3/5					
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5					
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5					
	August 9th	93			5/5					
Truth or Consequences, NM	August 27th	98	4,242 ft	7,289						
					<table border="1"> <tr> <td>Tess</td><td>5/5</td></tr> <tr> <td>Tony</td><td>4/5</td></tr> </table>		Tess	5/5	Tony	4/5
Tess	5/5									
Tony	4/5									
Why, AZ	August 18th	104	860 ft	480	3/5					

# 14

## html forms

### Getting Interactive

**So far all your web communication has been one-way: from your page to your visitors.** Golly, wouldn't it be nice if your visitors could talk back? That's where HTML forms come in: once you enable your pages with forms (along with a little help from a web server), your pages are going to be able to gather customer feedback, take an online order, get the next move in an online game, or collect the votes in a "hot or not" contest. In this chapter you're going to meet a whole team of HTML elements that work together to create web forms. You'll also learn a bit about what goes on behind the scenes in the server to support forms, and we'll even talk about keeping those forms stylish.



How forms work	646
What you write in HTML	648
What the browser creates	649
How the <form> element works	650
Getting ready to build the Bean Machine form	660
Adding the <form> element	661
How form element names work	662
Back to getting those <input> elements into your HTML	664
Adding some more input elements to your form	665
Adding the <select> element	666
Give the customer a choice of whole or ground beans	668
Punching the radio buttons	669
Using more input types	670
Adding the number and date input types	671
Completing the form	672
Adding the checkboxes and text area	673
Watching GET in action	679
Getting the form elements into HTML structure	684
Styling the form with CSS	686
A word about accessibility	688
What more could possibly go into a form?	689
Exercise Solutions	693

## 15

appendix: leftovers

**The Top Ten Topics (We Didn't Cover)**

We covered a lot of ground, and you're almost finished with this book. We'll miss you, but before we let you go, we wouldn't feel right about sending you out into the world without a little more preparation. We can't possibly fit everything you'll need to know into this relatively short chapter. Actually, we *did* originally include everything you need to know about HTML and CSS (not already covered by the other chapters), by reducing the type point size to .00004. It all fit, but nobody could read it. So, we threw most of it away, and kept the best bits for this Top Ten appendix.



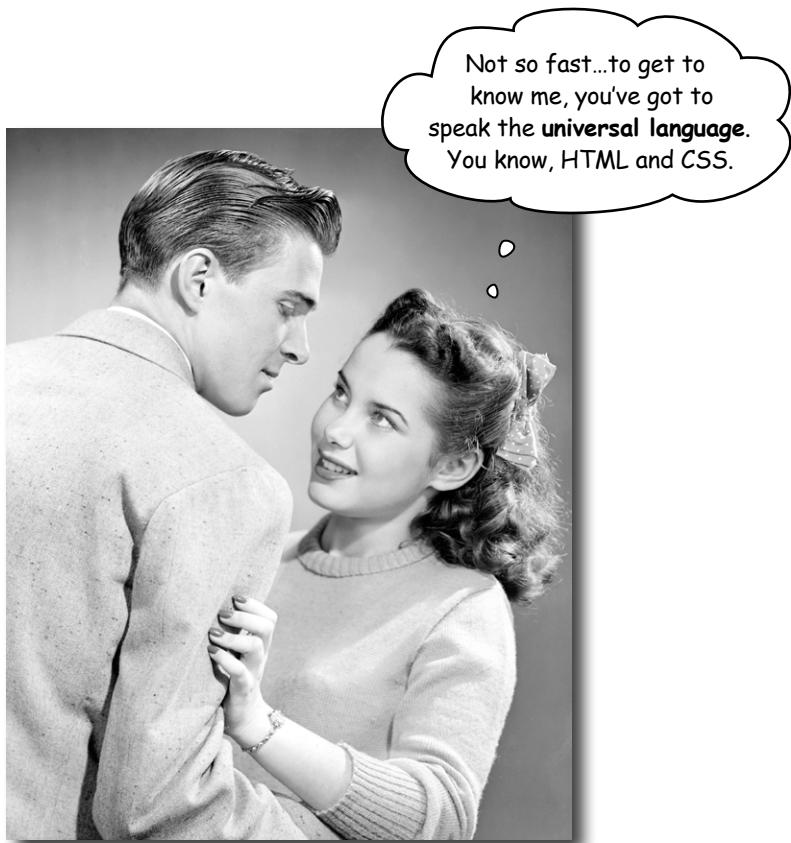
#1 More CSS selectors	698
#2 Vendor-specific CSS properties	700
#3 CSS transforms and transitions	701
#4 Interactivity	703
#5 HTML5 APIs and web apps	704
#6 More on Web Fonts	706
#7 Tools for creating web pages	707
#8 XHTML5	708
#9 Server-side scripting	709
#10 Audio	710

*i* **Index**

711

## 1 getting to know HTML

# The Language of the Web



**The only thing that is standing between you and getting yourself on the Web is learning to speak the lingo:**

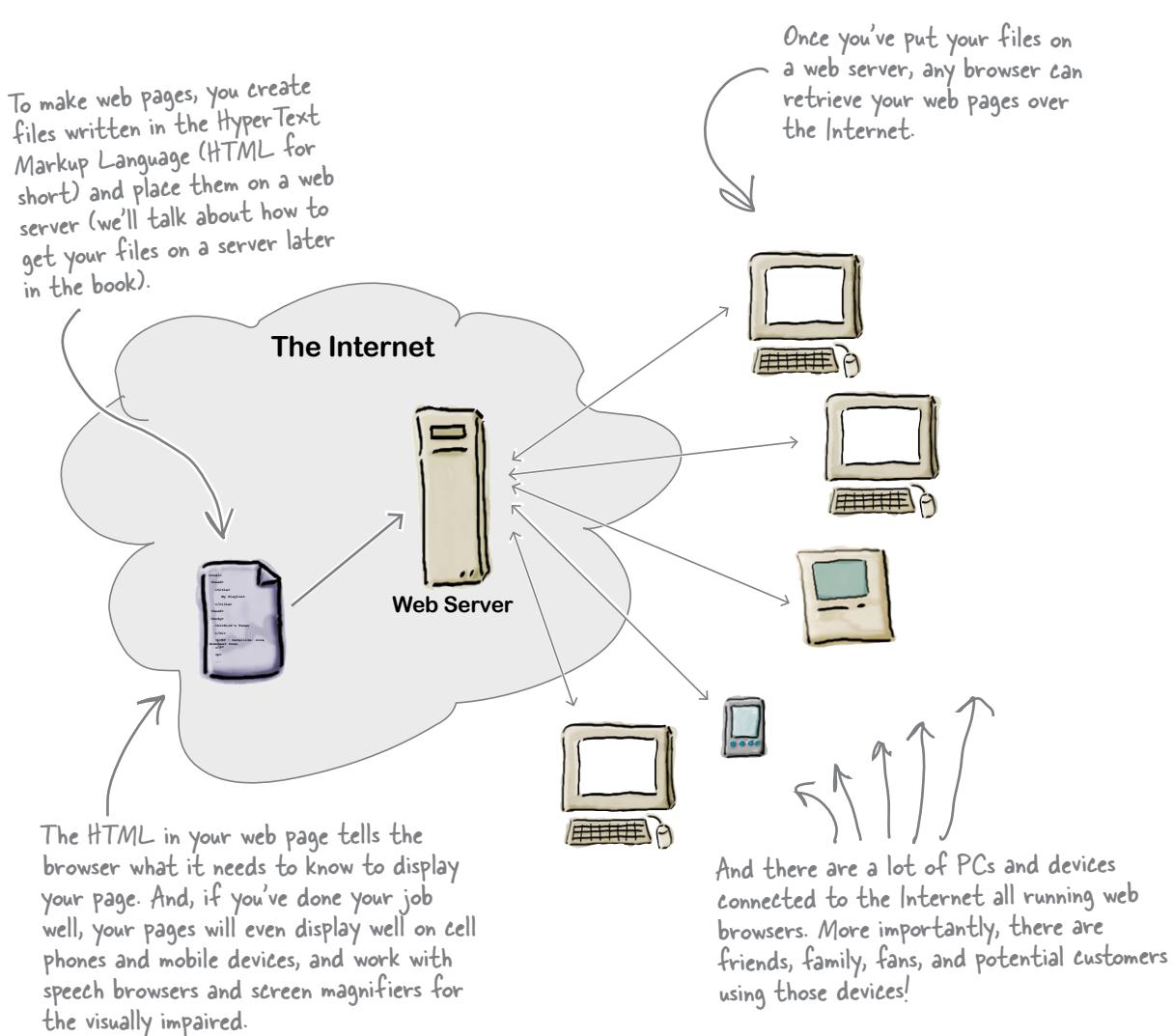
HyperText Markup Language, or HTML for short. So, get ready for some language lessons. After this chapter, not only are you going to understand some basic **elements** of HTML, but you'll also be able to speak HTML with a little **style**. Heck, by the end of this book you'll be talking HTML like you grew up in Webville.

# The Web

## ~~Video killed the radio star~~

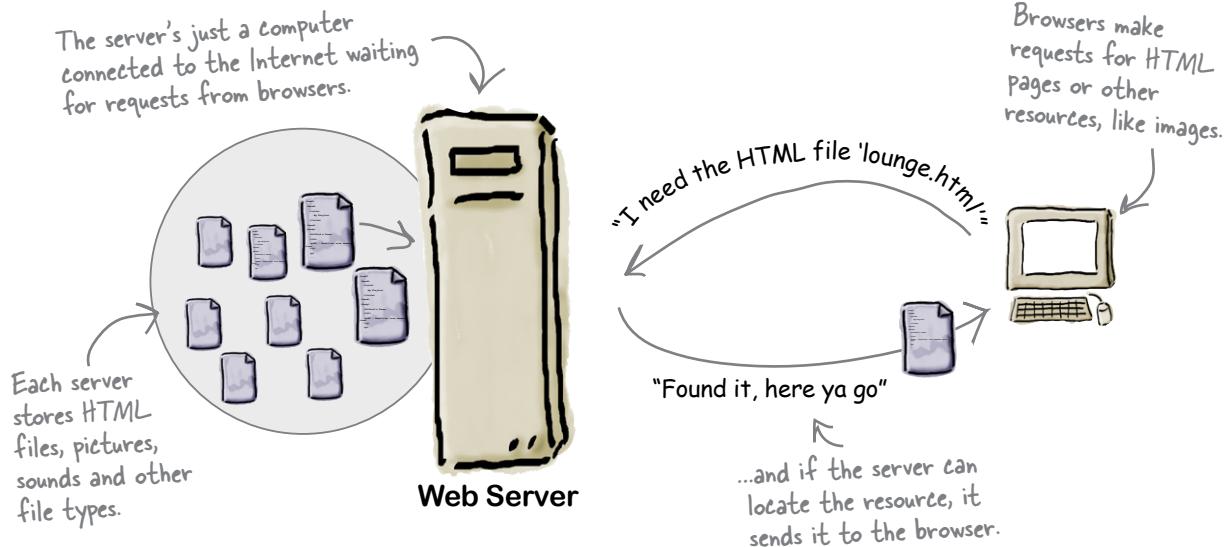
Want to get an idea out there? Sell something? Just need a creative outlet? Turn to the Web—we don't need to tell you it has become the universal form of communication. Even better, it's a form of communication **YOU** can participate in.

But if you really want to use the Web effectively, you've got to know a few things about **HTML**—not to mention, a few things about how the Web works too. Let's take a look from 30,000 feet:



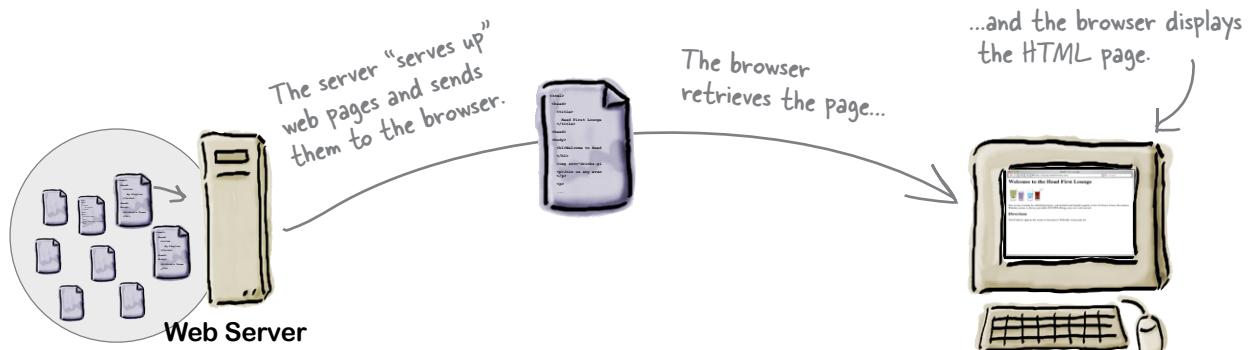
## What does the web server do?

Web servers have a full-time job on the Internet, tirelessly waiting for requests from web browsers. What kinds of requests? Requests for web pages, images, sounds, or maybe even a video. When a server gets a request for any of these resources, the server finds the resource, and then sends it back to the browser.



## What does the web browser do?

You already know how a browser works: you're surfing around the Web and you click on a link to visit a page. That click causes your browser to request an HTML page from a web server, retrieve it, and display the page in your browser window.



But how does the browser know how to display a page? That's where HTML comes in. HTML tells the browser all about the content and structure of the page. Let's see how that works...

# What you write (the HTML)

So, you know HTML is the key to getting a browser to display your pages, but what exactly does HTML look like? And what does it do?

Let's have a look at a little HTML...imagine you're going to create a web page to advertise the *Head First Lounge*, a local hangout with some good tunes, refreshing elixirs, and wireless access. Here's what you'd write in HTML:

```
<html>
  <head>
    <title>Head First Lounge</title> A
  </head>
  <body>
    <h1>Welcome to the Head First Lounge</h1> B
     C
    <p>
      D Join us any evening for refreshing elixirs,
      conversation and maybe a game or
      two of <em>Dance Dance Revolution</em>. E
      Wireless access is always provided;
      BYOWS (Bring your own web server).
    </p>
    <h2>Directions</h2> F
    <p>
      G You'll find us right in the center of
      downtown Webville. Come join us!
    </p>
  </body>
</html>
```



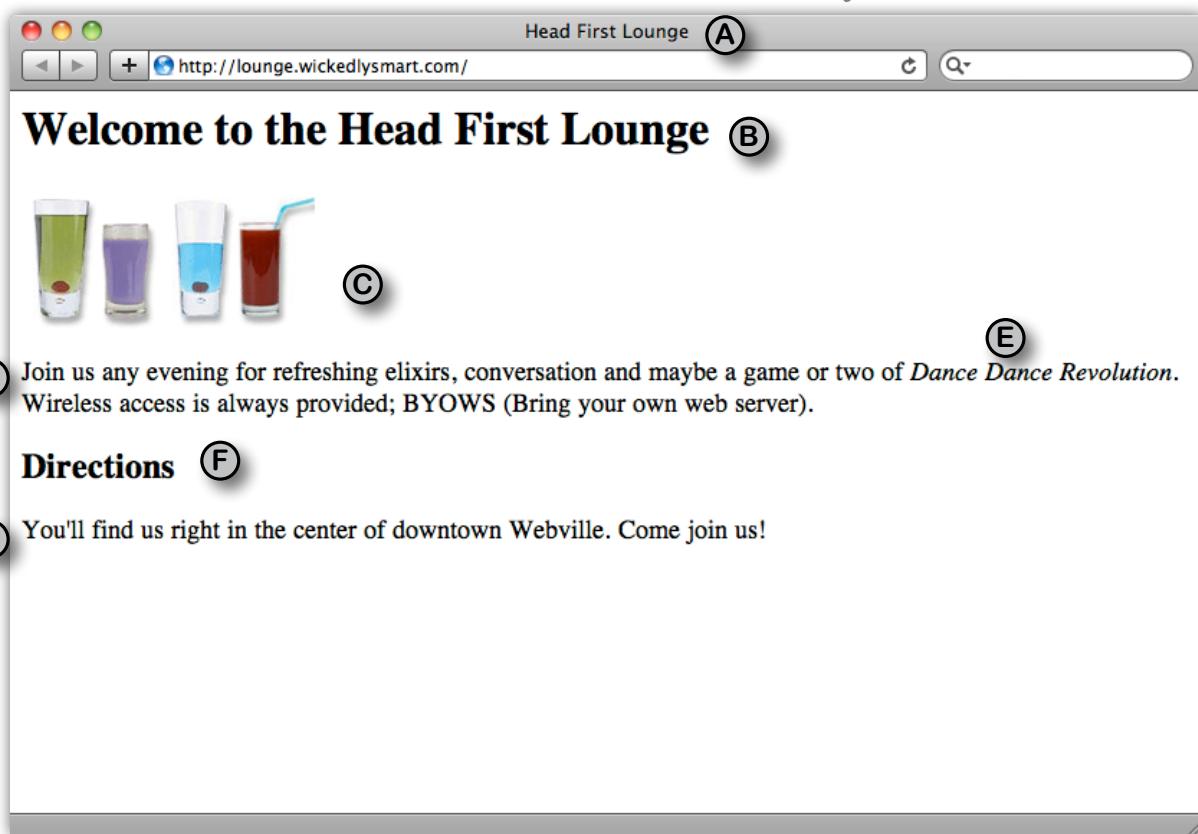
We don't expect you to know HTML yet.

At this point you should just be getting a feel for what HTML looks like; we're going to cover everything in detail in a bit. For now, study the HTML and see how it gets represented in the browser on the next page. Be sure to pay careful attention to each letter annotation and how and where it is displayed in the browser.

# What the browser creates

When the browser reads your HTML, it interprets all the *tags* that surround your text. Tags are just words or characters in angle brackets, like `<head>`, `<p>`, `<h1>`, and so on. The tags tell the browser about the *structure and meaning* of your text. So rather than just giving the browser a bunch of text, with HTML you can use tags to tell the browser what text is in a heading, what text is a paragraph, what text needs to be emphasized, or even where images need to be placed.

Let's check out how the browser interprets the tags in the Head First Lounge:



## *there are no Dumb Questions*

**Q:** So HTML is just a bunch of tags that I put around my text?

**A:** For starters. Remember that HTML stands for HyperText Markup Language, so HTML gives you a way to “mark up” your text with tags that tell the browser how your text is structured. But there is also the HyperText aspect of HTML, which we’ll talk about a little later in the book.

**Q:** How does the browser decide how to display the HTML?

**A:** HTML tells your browser about the structure of your document: where the headings are, where the paragraphs are, what text needs emphasis, and so on. Given this information, browsers have built-in default rules for how to display each of these elements.

But you don’t have to settle for the default settings. You can add your own style and formatting rules with CSS that determine font, colors, size, and a lot of other characteristics of your page. We’ll get back to CSS later in the chapter.

**Q:** The HTML for the Head First Lounge has all kinds of indentation and spacing, and yet I don’t see that when it is displayed in the browser. How come?

**A:** Correct, and good catch. Browsers ignore tabs, returns, and most spaces in HTML documents. Instead, they rely on your markup to determine where line and paragraph breaks occur.

So why did we insert our own formatting if the browser is just going to ignore it? To help us more easily read the document when we’re editing the HTML. As your

HTML documents become more complicated, you’ll find a few spaces, returns, and tabs here and there really help to improve the readability of the HTML.

**Q:** So there are two levels of headings, <h1> and a subheading <h2>?

**A:** Actually there are six, <h1> through <h6>, which the browser typically displays in successively smaller font sizes. Unless you are creating a complex and large document, you typically won’t use headings beyond <h3>.

**Q:** Why do I need the <html> tag? Isn’t it obvious this is an HTML document?

**A:** The <html> tag tells the browser your document is actually HTML. While some browsers will forgive you if you omit it, some won’t, and as we move toward “industrial-strength HTML” later in the book, you’ll see it is quite important to include this tag.

**Q:** What makes a file an HTML file?

**A:** An HTML file is a simple text file. Unlike a word processing file, there is no special formatting embedded in it. By convention, we add an “.html” to the end of the filename to give the operating system a better idea of what the file is. But, as you’ve seen, what really matters is what we put inside the file.

**Q:** Everyone is talking about HTML5. Are we using it? If so, why aren’t we saying “HTML-FIVE” instead of “HTML”?

**A:** You’re learning about HTML, and HTML5 just happens to be the latest version of HTML. HTML5 has had a lot of attention recently, and that’s because it simplifies

many of the ways we write HTML and enables some new functionality, which we’re going to cover in this book. It also provides some advanced features through its JavaScript application programming interfaces (APIs), and those are covered in Head First HTML5 Programming.

**Q:** Markup seems silly. What-you-see-is-what-you-get applications have been around since, what, the ’70s? Why isn’t the Web based on a format like Microsoft Word or a similar application?

**A:** The Web is created out of text files without any special formatting characters. This enables any browser in any part of the world to retrieve a web page and understand its contents. There are WYSIWYG applications out there like Dreamweaver, and they work great. But in this book we’re going to take it down to the bare metal, and start with text. Then you’re in good shape to understand what your Dreamweaver application is doing behind the scenes.

**Q:** Is there any way to put comments to myself in HTML?

**A:** Yes, if you place your comments in between <!-- and --> the browser will totally ignore them. Say you wanted to write a comment “Here’s the beginning of the lounge content.” You’d do that like this:

```
<!-- Here's the beginning of  
the lounge content -->
```

Notice that you can put comments on multiple lines. Keep in mind anything you put between the “<!--” and the “-->”, even HTML, will be ignored by the browser.



# Sharpen your pencil

You're closer to learning HTML than you think...

Here's the HTML for the Head First Lounge again. Take a look at the tags and see if you can guess what they tell the browser about the content. Write your answers in the space on the right; we've already done the first couple for you.

<html>	Tells the browser this is the start of HTML.
<head>	Starts the page "head" (more about this later)
<title>Head First Lounge</title>	
</head>	
<body>	
<h1>Welcome to the Head First Lounge</h1>	
<p>	
Join us any evening for refreshing elixirs, conversation and maybe a game or two of <em>Dance Dance Revolution</em>. Wireless access is always provided; BYOWS (Bring your own web server).	
</p>	
<h2>Directions</h2>	
<p>	
You'll find us right in the center of downtown Webville. Come join us!	
</p>	
</body>	
</html>	



## Sharpen your pencil

### Solution

```
<html>
  <head>
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the Head First Lounge</h1>
    
    <p>
      Join us any evening for refreshing elixirs,
      conversation and maybe a game or
      two of <em>Dance Dance Revolution</em>.
      Wireless access is always provided;
      BYOWS (Bring your own web server).
    </p>
    <h2>Directions</h2>
    <p>
      You'll find us right in the center of
      downtown Webville. Come join us!
    </p>
  </body>
</html>
```

Tells the browser this is the start of HTML.

Starts the page "head".

Gives the page a title.

End of the head.

Start of the body of page.

Tells browser that "Welcome to..." is a heading.

Places the image "drinks.gif" here.

Start of a paragraph.

Puts emphasis on Dance Dance Revolution.

End of paragraph.

Tells the browser that "Directions" is a subheading.

Start of another paragraph.

End of paragraph.

End of the body.

Tells the browser this is the end of the HTML.



## Your big break at Starbuzz Coffee

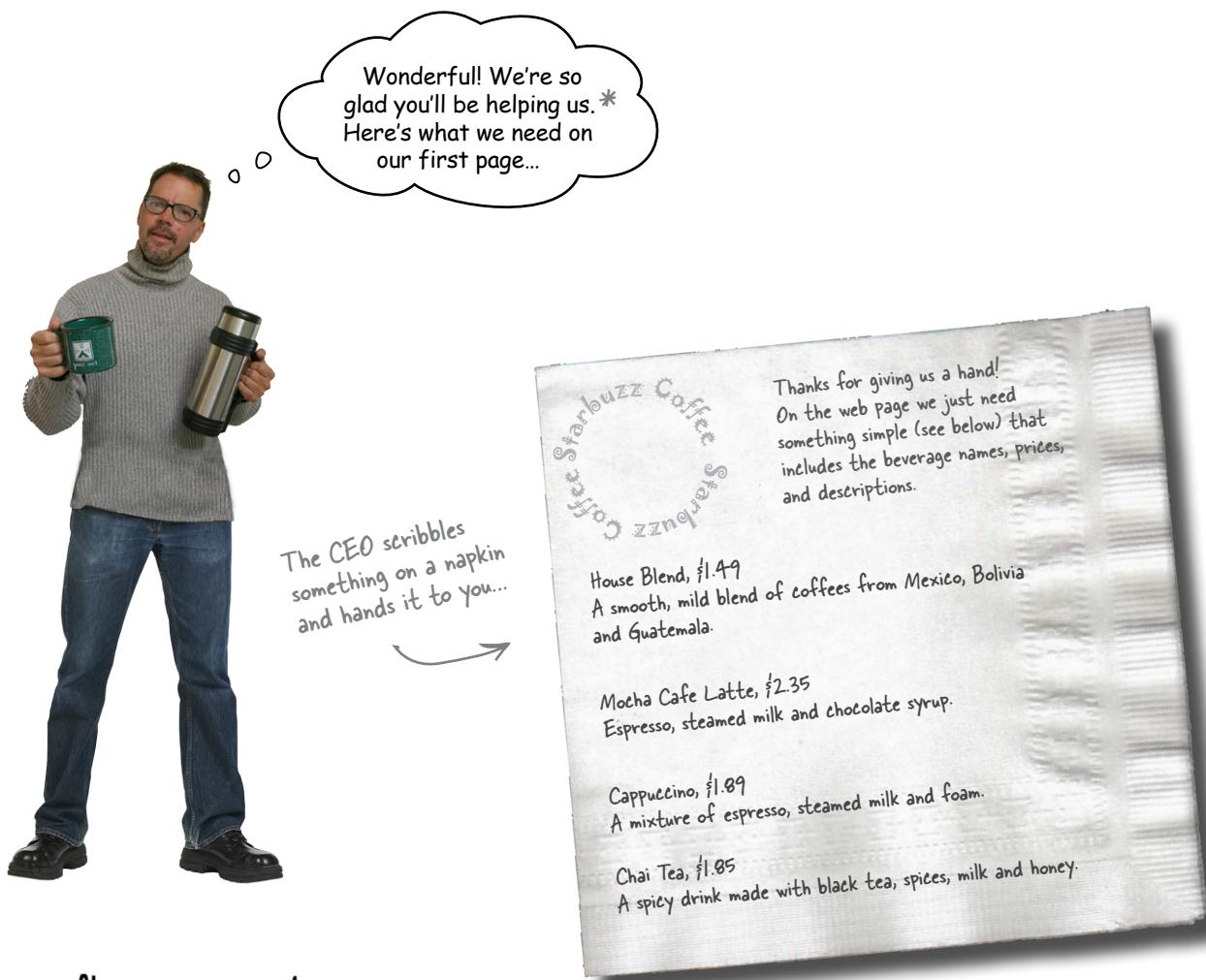
Starbuzz Coffee has made a name for itself as the fastest growing coffee shop around. If you've seen one on your local corner, look across the street—you'll see another one.

In fact, they've grown so quickly, they haven't even managed to put up a web page yet...and therein lies your big break: By chance, while buying your Starbuzz Chai Tea, you run into the Starbuzz CEO...



Decisions, decisions.  
Check your first priority below (choose only one):

- A. Give dog a bath.
- B. Finally get my checking account balanced.
- C. Take the Starbuzz gig and launch BIG-TIME web career.
- D. Schedule dentist appointment.



### Sharpen your pencil

Take a look at the napkin. Can you determine the *structure* of it? In other words, are there obvious headings? Paragraphs? Is it missing anything like a title?

Go ahead and mark up the napkin (using your pencil) with any structure you see, and add anything that is missing.

You'll find our answers at the end of Chapter 1.

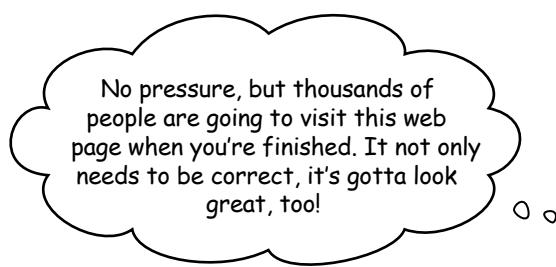
\* If by chance you chose option A, B, or D on the previous page, we recommend you donate this book to a good library, use it as kindling this winter, or what the heck, go ahead and sell it on Amazon and make some cash.

# Creating the Starbuzz web page

**Of course, the only problem with all this is that you haven't actually created any web pages yet. But that's why you decided to dive head first into HTML, right?**

**No worries, here's what you're going to do on the next few pages:**

- ① Create an HTML file using your favorite text editor.**
- ② Type in the menu the Starbuzz CEO wrote on the napkin.**
- ③ Save the file as “index.html”.**
- ④ Open the file “index.html” in your favorite browser, step back, and watch the magic happen.**



# Creating an HTML file (Mac)

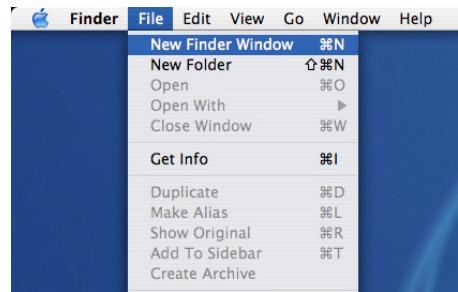
All HTML files are text files. To create a text file, you need an application that allows you to create plain text without throwing in a lot of fancy formatting and special characters. You just need plain, pure text.

We'll use TextEdit on the Mac in this book; however, if you prefer another text editor, that should work fine as well. And, if you're running Windows, you'll want to skip ahead a couple of pages to the Windows instructions.

## Step one:

Navigate to your **Applications** folder

The TextEdit application is in the *Applications* folder. The easiest way to get there is to choose New Finder Window from the Finder's File menu and then look for the Application directly in your shortcuts. When you've found it, click on Applications.



## Step two:

Locate and run **TextEdit**

You'll probably have lots of applications listed in your *Applications* folder, so scroll down until you see TextEdit. To run the application, double-click on the TextEdit icon.



## Step three (optional):

Keep **TextEdit** in your Dock

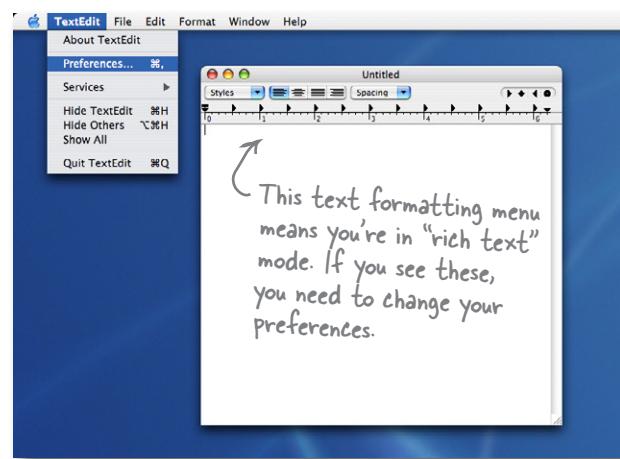
If you want to make your life easier, click and hold on the TextEdit icon in the Dock (this icon appears once the application is running). When it displays a pop-up menu, choose Options, then "Keep in Dock." That way, the TextEdit icon will always appear in your Dock and you won't have to hunt it down in the *Applications* folder every time you need to use it.



## Step four:

### Change your TextEdit Preferences

By default, TextEdit is in “rich text” mode, which means it will add its own formatting and special characters to your file when you save it—not what you want. So, you’ll need to change your TextEdit Preferences so that TextEdit saves your work as a pure text file. To do this, first choose the Preferences menu item from the TextEdit menu.



## Step five:

### Set Preferences for Plain text

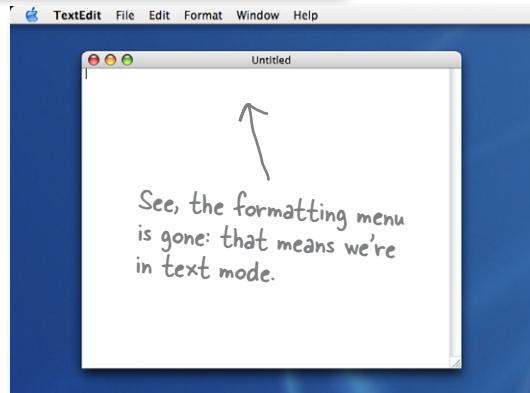
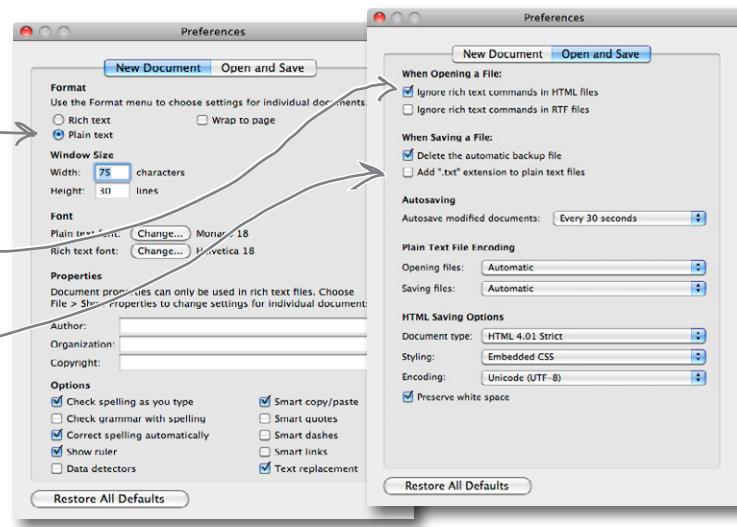
Once you see the Preferences dialog box, there are three things you need to do.

First, choose “Plain text” as the default editor mode in the New Document tab.

In the “Open and Save” tab, make sure “Ignore rich text commands in HTML files” is checked.

Last, make sure that the “Add .txt extension to plain text files” is **unchecked**.

That's it; to close the dialog box, click on the red button in the top-left corner.



## Step six:

### Quit and restart

Now quit out of TextEdit by choosing Quit from the TextEdit menu, and then restart the application. This time, you’ll see a window with no fancy text formatting menus at the top. You’re now ready to create some HTML.

# Creating an HTML file (Windows)

If you're reading this page you must be a Windows 7 user. If you're not, you might want to skip a couple of pages ahead. Or, if you just want to sit in the back and not ask questions, we're okay with that too.

To create HTML files in Windows 7, we're going to use Notepad—it ships with every copy of Windows, the price is right, and it's easy to use. If you've got your own favorite editor that runs on Windows 7, that's fine too; just make sure you can create a plain-text file with an ".html" extension.

Assuming you're using Notepad, here's how you're going to create your first HTML file.

## Step one:

Open the **Start** menu and navigate to Notepad.

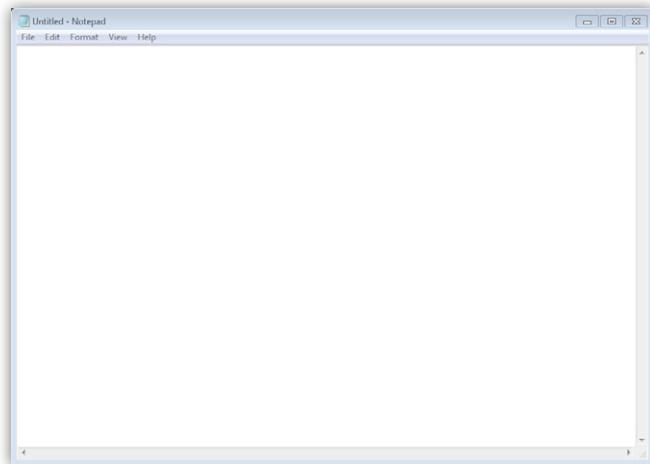
You'll find the Notepad application in *Accessories*. The easiest way to get there is to click on the Start menu, then on All Programs, then Accessories. You'll see Notepad listed there.



## Step two:

### Open Notepad.

Once you've located Notepad in the *Accessories* folder, go ahead and click on it. You'll see a blank window ready for you to start typing HTML.



*But recommended*



## Step three (optional):

### Don't hide extensions of well-known file types.

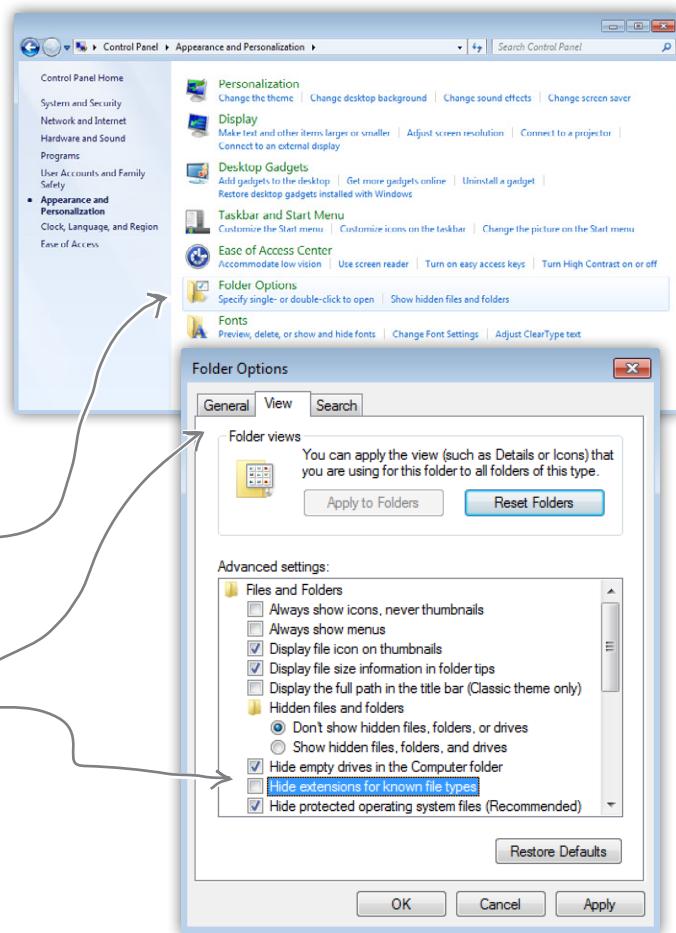
By default, Windows File Explorer hides the file extensions of well-known file types. For example, a file named "Irule.html" will be shown in the Explorer as "Irule" without its ".html" extension.

It's much less confusing if Windows shows you these extensions, so let's change your folder options so you can see them.

First, open Folder Options by clicking the Start button, clicking Control Panel, clicking "Appearance and Personalization," and then clicking Folder Options.

Next, in the View tab, under "Advanced settings," scroll down until you see "Hide extensions for known file types" and *uncheck* this option.

That's it. Click on the OK button to save the preference and you'll now see the file extensions in the Explorer.



# <sup>there are no</sup> Dumb Questions

**Q:** Why am I using a simple text editor?  
Aren't there powerful tools like Dreamweaver  
and Expression Web for creating web pages?

**A:** You're reading this book because you want to understand the true technologies used for web pages, right? Now those are all great tools, but they do a lot of the work for you, and until you are a master of HTML and CSS, you want to learn this stuff without a big tool getting in your way.

Once you're a master, however, these tools do provide some nice features like syntax checking and previews. At that point, when you view the "code" window, you'll understand everything in it, and you'll find that changes to the raw HTML and CSS are often a lot faster than going through a user interface. You'll also find that as standards change, these tools aren't always updated right away and may not support the most recent standards until their next release cycle. Since you'll know how to change the HTML and CSS without the tool, you'll be able to keep up with the latest and greatest all the time.

There are many more fully featured editors that include great features like clips (for automatically inserting bits of HTML you write often), preview (for previewing directly in the editor before you test in the browser), syntax coloring (so tags are a different color from content), and much more. Once you get the hang of writing basic HTML and CSS in a simple editor, it may be worth checking out one of the fancier editors, such as Coda, TextMate, CoffeeCup, or Aptana Studio. There are many out there to choose from (both free and not).

**Q:** I get the editor, but what browser am I supposed to be using? There are so many—Internet Explorer, Chrome, Firefox, Opera, Safari—what's the deal?

**A:** The simple answer: use whatever browser you like. HTML and CSS are industry standards, which means that all browsers try to support HTML and CSS in the same way (just make sure you are using the newest version of the browser for the best support).

The complex answer: in reality there are slight differences in the way browsers handle your pages. If you've got users who will be accessing your pages in a variety of browsers, then always test your web page in several different browsers. Some pages will look exactly the same; some won't. The more advanced you become with HTML and CSS, the more these slight differences may matter to you, and we'll get into some of these subtleties throughout the book.

Any of the major browsers—Internet Explorer, Chrome, Firefox, Opera, and Safari—will work for most examples (except where noted); they are all modern browsers with great HTML and CSS support. And as a web developer, you'll be expected to test your code in more than one browser, so we encourage you to download and get familiar with at least two!

**Q:** I'm creating these files on my own computer—how am I going to view these on the Web?

**A:** That's one great thing about HTML: you can create files and test them on your own computer and then later publish them on the Web. Right now, we're going to worry about how to create the files and what goes in them. We'll come back to getting them on the Web a bit later.



## Meanwhile, back at Starbuzz Coffee...

Okay, now that you know the basics of creating a plain-text file, you just need to get some content into your text editor, save it, and then load it into your browser.

Start by typing in the beverages straight from the CEO's napkin; these beverages are the content for your page. You'll be adding some HTML markup to give the content some structure in a bit, but for now, just get the basic content typed in. While you're at it, go ahead and add "Starbuzz Coffee Beverages" at the top of the file.

Type in the info from  
the napkin like this.



Untitled - Notepad

File Edit Format View Help

Starbuzz Coffee Beverages

House Blend, \$1.49  
A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.

Mocha Cafe Latte, \$2.35  
Espresso, steamed milk and chocolate syrup.

Cappuccino, \$1.89  
A mixture of espresso, steamed milk and foam.

Chai Tea, \$1.85  
A spicy drink made with black tea, spices, milk and honey.|

Untitled.txt

Starbuzz Coffee Beverages

House Blend, \$1.49  
A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.

Mocha Cafe Latte, \$2.35  
Espresso, steamed milk and chocolate syrup.

Cappuccino, \$1.89  
A mixture of espresso, steamed milk and foam.

Chai Tea, \$1.85  
A spicy drink made with black tea, spices, milk and honey.

Mac

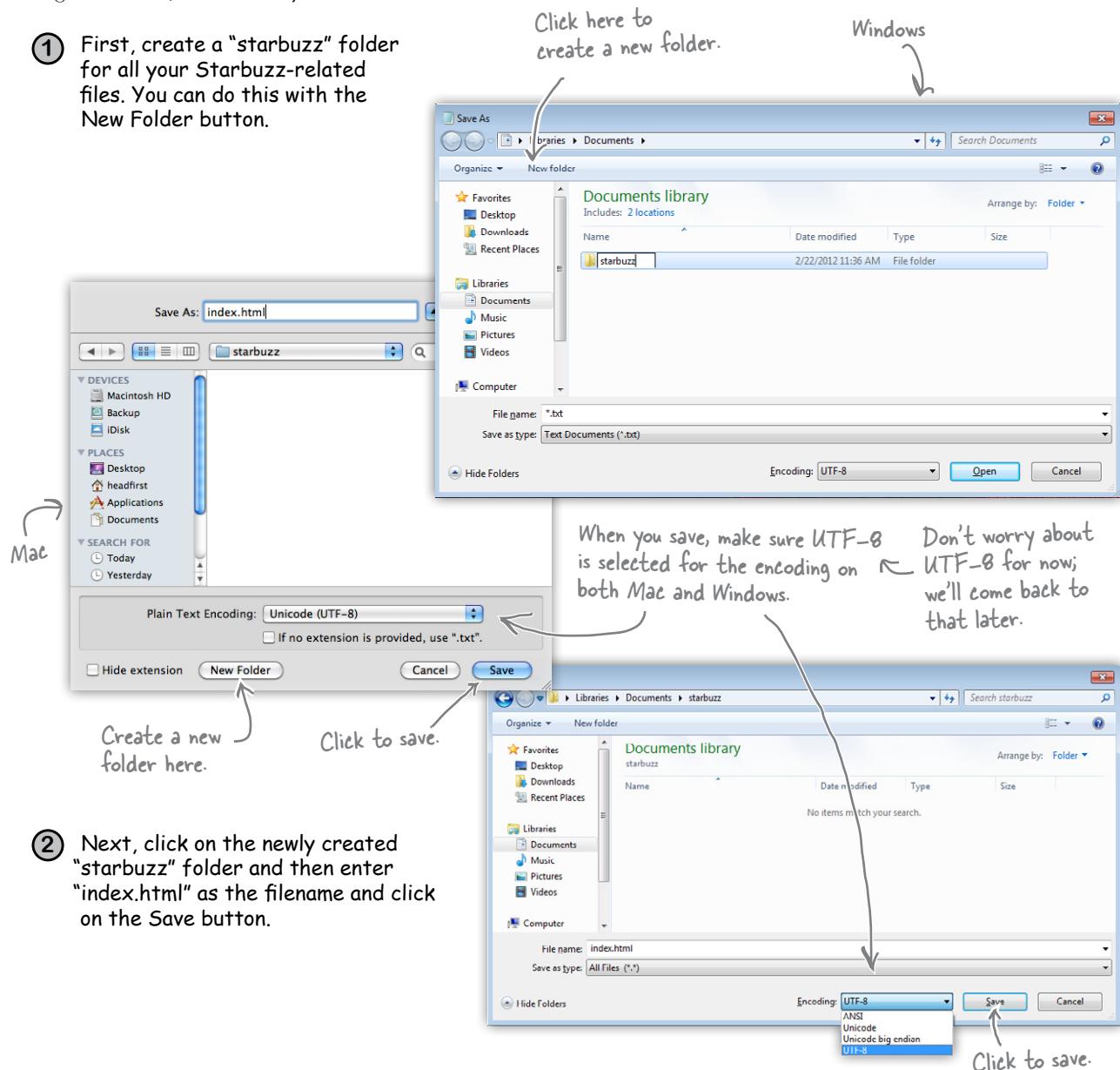
Windows

# Saving your work

Once you've typed in the beverages from the CEO's napkin, you're going to save your work in a file called "index.html". Before you do that, you'll want to create a folder named "starbuzz" to hold the site's files.

To get this all started, choose Save from the File menu and you'll see a Save As dialog box. Then, here's what you need to do:

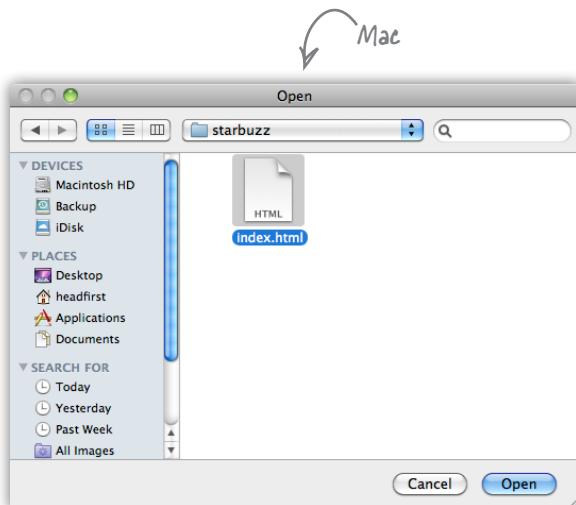
- ① First, create a "starbuzz" folder for all your Starbuzz-related files. You can do this with the New Folder button.



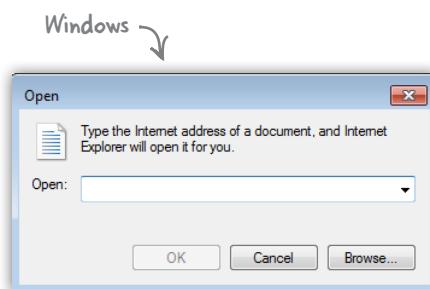
- ② Next, click on the newly created "starbuzz" folder and then enter "index.html" as the filename and click on the Save button.

# Opening your web page in a browser

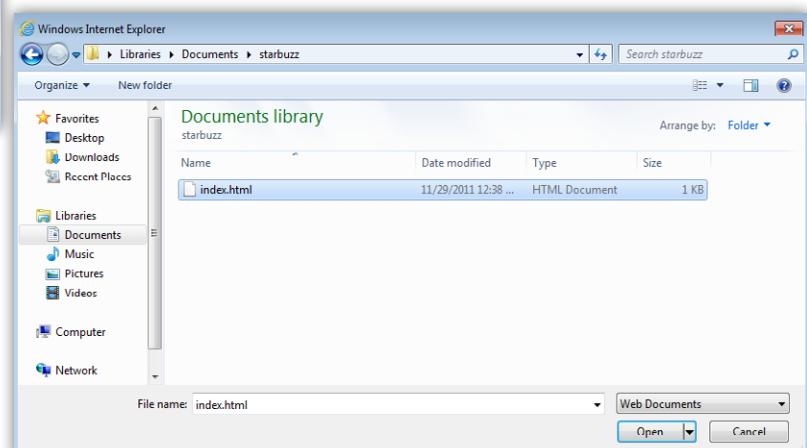
Are you ready to open your first web page? Using your favorite browser, choose “Open File...” (or “Open...” using Windows 7 and Internet Explorer) from the File menu and navigate to your “index.html” file. Select it and click Open.



On the Mac, navigate to your file, and select it by clicking on the file icon and then on the Open button.



In Windows Internet Explorer it's a two-step process. First, you'll get the Open dialog box.

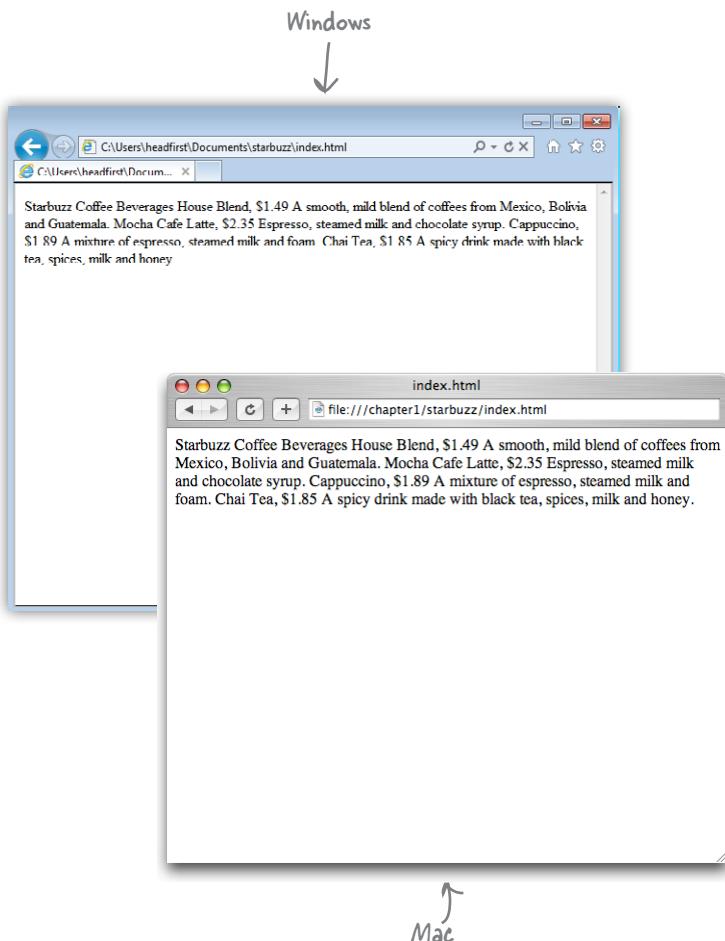


Then click Browse to get a browse dialog and navigate to where you saved your file.

## Take your page for a test drive



Success! You've got the page loaded in the browser, although the results are a little...uh...unsatisfying. But that's just because all you've done so far is go through the mechanics of creating a page and viewing it in the browser. And so far, you've only typed in the *content* of the web page. That's where HTML comes in. HTML gives you a way to tell the browser about the *structure* of your page. What's structure? As you've already seen, it is a way of marking up your text so that the browser knows what's a heading, what text is in a paragraph, what text is a subheading, and so on. Once the browser knows a little about the structure, it can display your page in a more meaningful and readable manner.



Depending on your operating system and browser, often you can just double-click the HTML file or drag it on top of the browser icon to open it. Much simpler.



# Markup Magnets

So, let's add that structure...

Your job is to add structure to the text from the Starbuzz napkin. Use the fridge magnets at the bottom of the page to mark up the text so that you've indicated which parts are headings, subheadings and paragraph text. We've already done a few to get you started. You won't need all the magnets below to complete the job; some will be left over.

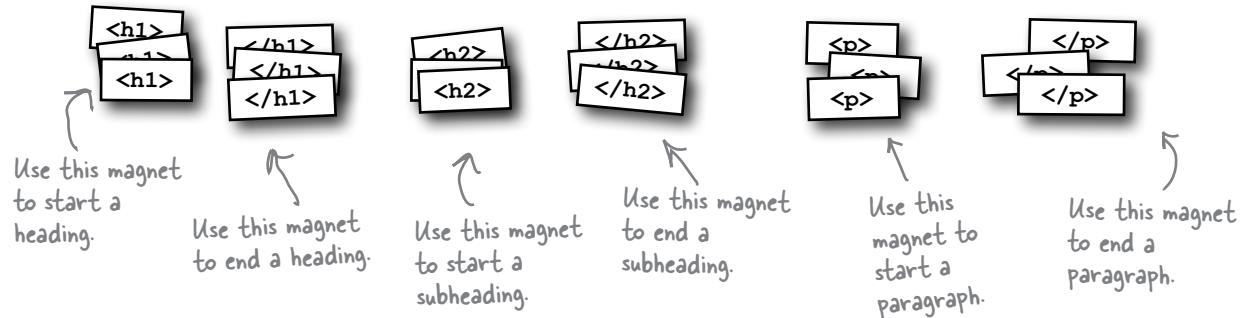
```
<h1> Starbuzz Coffee Beverages </h1>

House Blend, $1.49
A smooth, mild blend of coffees from Mexico, Bolivia and
Guatemala.

<h2> Mocha Cafe Latte, $2.35 </h2>
<p> Espresso, steamed milk and chocolate syrup. </p>

Cappuccino, $1.89
A mixture of espresso, steamed milk and foam.

Chai Tea, $1.85
A spicy drink made with black tea, spices, milk and honey.
```





## Congratulations, you've just written your first HTML!

They might have looked like fridge magnets, but you were really *marking up* your text with HTML. Only, as you know, we usually refer to the magnets as *tags*.

Check out the markup below and compare it to your magnets on the previous page.

```
<h1>Starbuzz Coffee Beverages</h1>
```

Use the `<h1>` and `</h1>` tags to mark headings. All the text in between is the actual content of the heading.

```
<h2>House Blend, $1.49</h2>
```

```
<p>A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.</p>
```

```
<h2>Mocha Cafe Latte, $2.35</h2>
```

```
<p>Espresso, steamed milk and chocolate syrup.</p>
```

```
<h2>Cappuccino, $1.89</h2>
```

```
<p>A mixture of espresso, steamed milk and foam.</p>
```

```
<h2>Chai Tea, $1.85</h2>
```

```
<p>A spicy drink made with black tea, spices, milk and honey.</p>
```

The `<h2>` and `</h2>` tags go around a subheading. Think of an `<h2>` heading as a subheading of an `<h1>` heading.

The `<p>` and `</p>` tags go around a block of text that is a paragraph. That can be one or many sentences.

Notice that you don't have to put matching tags on the same line. You can put as much content as you like between them.

# Are we there yet?

You have an HTML file with markup—does that make a web page? Almost. You've already seen the `<html>`, `<head>`, `<title>`, and `<body>` tags, and we just need to add those to make this a first-class HTML page...

First, surround your HTML with `<html>` & `</html>` tags. This tells the browser the content of the file is HTML.

```
<html>
```

```
<head>
    <title>Starbuzz Coffee</title>
</head>
```

Next add `<head>` and `</head>` tags. The head contains information about your web page, like its title. For now, think about it this way: the head allows you to tell the browser things about the web page.

Go ahead and put a title inside the head. The title usually appears at the top of the browser window.

The head consists of the `<head>` & `</head>` tags and everything in between.

```
<body>
```

```
<h1>Starbuzz Coffee Beverages</h1>
<h2>House Blend, $1.49</h2>
<p>A smooth, mild blend of coffees from Mexico,
    Bolivia and Guatemala.</p>

<h2>Mocha Cafe Latte, $2.35</h2>
<p>Espresso, steamed milk and chocolate syrup.</p>

<h2>Cappuccino, $1.89</h2>
<p>A mixture of espresso, steamed milk and foam.</p>

<h2>Chai Tea, $1.85</h2>
<p>A spicy drink made with black tea, spices,
    milk and honey.</p>
```

```
</body>
```

```
</html>
```

The body contains all the content and structure of your web page—the parts of the web page that you see in your browser.



Keep your head  
and body separate  
when writing HTML.

## Another test drive



**Go ahead and change your “index.html” file by adding in the `<head>`, `</head>`, `<title>`, `</title>`, `<body>` and `</body>` tags. Once you’ve done that, save your changes and reload the file into your browser.**

You can reload the index.html file by selecting the Open File menu item again, or by using your browser’s reload button.

Notice that the title, which you specified in the `<head>` element, shows up here.

The screenshot shows a web browser window with the title "Starbuzz Coffee". The address bar indicates the file is located at "file:///chapter1/starbuzz/index.html". The main content area displays the following text:

**Starbuzz Coffee Beverages**

**House Blend, \$1.49**  
A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.

**Mocha Cafe Latte, \$2.35**  
Espresso, steamed milk and chocolate syrup.

**Cappuccino, \$1.89**  
A mixture of espresso, steamed milk and foam.

**Chai Tea, \$1.85**  
A spicy drink made with black tea, spices, milk and honey.

Now things look a bit better.  
The browser has interpreted  
your tags and created a  
display for the page that  
is not only more structured,  
but also more readable.



# Tags dissected

**Okay, you've seen a bit of markup, so let's zoom in and take a look at how tags really work.**



You usually put tags around some piece of content. Here we're using tags to tell the browser that our content, "Starbuzz Coffee Beverages," is a top-level heading (that is, heading level one).

Here's the opening tag that begins the heading.

Tags consist of the tag name surrounded by angle brackets; that is, the < and > characters.

<**h1**> Starbuzz Coffee Beverages </**h1**>

The whole shebang is called an element. In this case, we can call it the **<h1>** element. An element consists of the enclosing tags and the content in between.

This is the closing tag that ends the heading; in this case the </**h1**> tag is ending an **<h1>** heading. You know it's a closing tag because it comes after the content, and it's got a "/" before the "h1". All closing tags have a "/" in them.

We call an opening tag and its closing tag matching tags.

**To tell the browser about the structure of your page, use pairs of tags around your content.**

**Remember:**

**Element = Opening Tag + Content + Closing Tag**

## there are no Dumb Questions

**Q:** So matching tags don't have to be on the same line?

**A:** No; remember the browser doesn't really care about tabs, returns, and most spaces. So, your tags can start and end anywhere on the same line, or they can start and end on different lines. Just make sure you start with an opening tag, like `<h2>`, and end with a closing tag, like `</h2>`.

**Q:** Why do the closing tags have that extra "/"?

**A:** That "/" in the closing tag is to help both you and the browser know where a particular piece of structured content ends. Otherwise, the closing tags would look just like the opening tags, right?

**Q:** I've noticed the HTML in some pages doesn't always match opening tags with closing tags.

**A:** Well, the tags are supposed to match. In general, browsers do a pretty good job of figuring out what you mean if you write incorrect HTML. But, as you're going to see, these days there are big benefits to writing totally correct HTML. If you're worried you'll never be able to write perfect HTML, don't be; there are plenty of tools to verify your code before you put it on a web server so the whole world can see it. For now, just get in the habit of always matching your opening tags with closing tags.

**Q:** Well, what about that `` tag in the lounge example? Did you forget the closing tag?

**A:** Wow, sharp eye. There are some elements that use a shorthand notation with only one tag. Keep that in the back of your mind for now, and we'll come back to it in a later chapter.

**Q:** An element is an opening tag + content + closing tag, but can't you have tags inside other tags? Like the `<head>` and `<body>` are inside an `<html>` tag?

**A:** Yes, HTML tags are often "nested" like that. If you think about it, it's natural for an HTML page to have a body, which contains a paragraph, and so on. So many HTML elements have other HTML elements between their tags. We'll take a good look at this kind of thing in later chapters, but for now just get your mind noticing how the elements relate to each other in a page.



Tags can be a little more interesting than what you've seen so far. Here's the paragraph tag with a little extra added to it. What do you think this does?

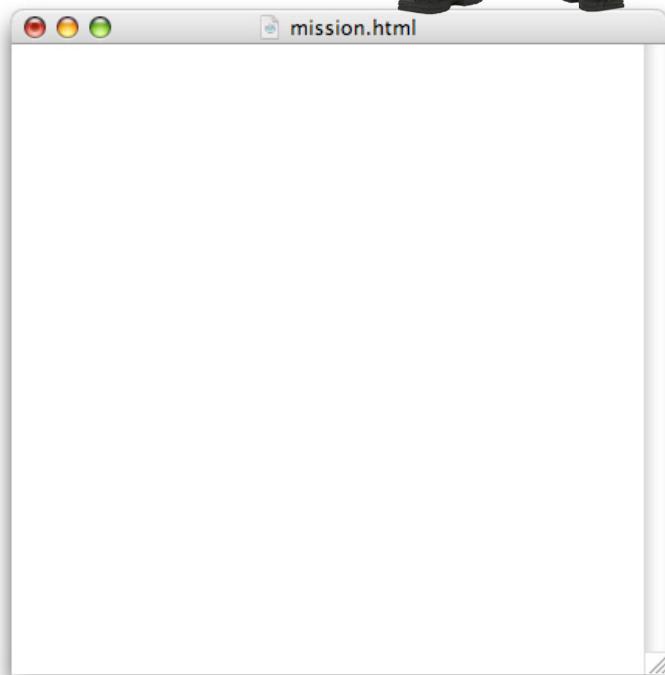
```
<p id="houseblend">A smooth, mild  
blend of coffees from Mexico, Bolivia  
and Guatemala.</p>
```



## Exercise



Oh, I forgot to mention, we need our company mission on a page, too. Grab the mission statement off one of our coffee cups and create another page for it...



- ➊ Write the HTML for the new "mission.html" page here.
- ➋ Type in your HTML using a text editor, and save it as "mission.html" in the same folder as your "index.html" file.
- ➌ Once you've done that, open "mission.html" in your browser.
- ➍ Check your work at the end of the chapter before moving on...



Okay, it looks like you're getting somewhere. You've got the main page and the mission page all set. But don't forget the CEO said the site needs to look great too. Don't you think it needs a little style?

**Right. We have the structure down, so now we're going to concentrate on its presentation.**

You already know that HTML gives you a way to describe the structure of the content in your files. When the browser displays your HTML, it uses its own built-in default style to present this structure. But relying on the browser for style obviously isn't going to win you any "designer of the month" awards.

That's where CSS comes in. CSS gives you a way to describe how your content should be presented. Let's get our feet wet by creating some CSS that makes the Starbuzz page look a little more presentable (and launch your web career in the process).

CSS is an abbreviation for Cascading Style Sheets. We'll get into what that all means later, but for now just know that CSS gives you a way to tell the browser how elements in your page should look.

# Meet the style element

To add style, you add a new (say it with us) E-L-E-M-E-N-T to your page—the `<style>` element. Let's go back to the main Starbuzz page and add some style. Check it out...

```

<html>
  <head>
    <title>Starbuzz Coffee</title>
    <style type="text/css">
      </style>
  </head>
  <body>
    <h1>Starbuzz Coffee Beverages</h1>

    <h2>House Blend, $1.49</h2>
    <p>A smooth, mild blend of coffees from Mexico, Bolivia and
Guatemala.</p>

    <h2>Mocha Caffe Latte, $2.35</h2>
    <p>Espresso, steamed milk and chocolate syrup.</p>

    <h2>Cappuccino, $1.89</h2>
    <p>A mixture of espresso, steamed milk and milk foam.</p>

    <h2>Chai Tea, $1.85</h2>
    <p>A spicy drink made with black tea, spices, milk and honey.</p>
  </body>
</html>

```

The `<style>` element is placed inside the head of your HTML.

Just like other elements, the `<style>` element has an opening tag, `<style>`, and a closing tag, `</style>`.

The `<style>` tag also has an (optional) attribute, called `type`, which tells the browser the kind of style you're using. Because you're going to use CSS, you can specify the "text/css" type.

And here's where you're going to define the styles for the page.

---

*there are no Dumb Questions*

---

**Q:** An element can have an “attribute”? What does that mean?

**A:** Attributes give you a way to provide additional information about an element. Like, if you have a `<style>` element, the attribute allows you to say exactly what kind of style you're talking about. You'll be seeing a lot more attributes for various elements; just remember they give you some extra info about the element.

**Q:** Why do I have to specify the type of the style (“text/css”) as an attribute of the style? Are there other kinds of style?

**A:** At one time the designers of HTML thought there would be other styles, but as it turns out we've all come to our senses since then and you can just use `<style>` without an attribute—all the browsers will know you mean CSS. We're disappointed; we were holding our breath for the `<style type="50sKitsch">` style. Oh well.

## Giving Starbuzz some style...

Now that you've got a `<style>` element in the HTML head, all you need to do is supply some CSS to give the page a little pizzazz. Below you'll find some CSS already "baked" for you. Whenever you see the  logo, you're seeing HTML and CSS that you should type in as-is. *Trust us.* You'll learn how the markup works later, after you've seen what it can do.

So, take a look at the CSS and then add it to your "index.html" file. Once you've got it typed in, save your file.

```
<html>
  <head>
    <title>Starbuzz Coffee</title>
    <style type="text/css">
      body {
        background-color: #d2b48c;
        margin-left: 20%;
        margin-right: 20%;
        border: 2px dotted black;
        padding: 10px 10px 10px 10px;
        font-family: sans-serif;
      }
    </style>
  </head>

  <body>
    <h1>Starbuzz Coffee Beverages</h1>

    <h2>House Blend, $1.49</h2>
    <p>A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.</p>

    <h2>Mocha Caffe Latte, $2.35</h2>
    <p>Espresso, steamed milk and chocolate syrup.</p>

    <h2>Cappuccino, $1.89</h2>
    <p>A mixture of espresso, steamed milk and milk foam.</p>

    <h2>Chai Tea, $1.85</h2>
    <p>A spicy drink made with black tea, spices, milk and honey.</p>
  </body>
</html>
```



Ready Bake  
CSS

CSS uses a syntax that is totally different from HTML.

# Cruisin' with style...



It's time for another test drive, so reload your "index.html" file again. This time, you'll see the Starbuzz web page has a whole new look.

Background color is now tan.

Now we have margins around the content

We've got a black dotted border around the content.

There's now some padding between the content and the border (on all sides).

We're using a different font for a cleaner look.

## Starbuzz Coffee Beverages

### House Blend, \$1.49

A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.

### Mocha Caffe Latte, \$2.35

Espresso, steamed milk and chocolate syrup.

### Cappuccino, \$1.89

A mixture of espresso, steamed milk and milk foam.

### Chai Tea, \$1.85

A spicy drink made with black tea, spices, milk and honey.

Margin



### Watch it!

If you're using IE, you might not see the border.

Internet Explorer does not display the border around the body correctly. Try loading the page in Firefox, Chrome or Safari to see the border.

Whoa! Very nice. We're in business now!



## WHO DOES WHAT?

Even though you've just glanced at CSS, you've already begun to see what it can do. Match each line in the style definition to what it does.

`background-color: #d2b48c;`

Defines the font to use for text.

`margin-left: 20%;`

Defines a border around the body that is dotted and the color black.

`border: 2px dotted black;`

Sets the left and right margins to take up 20% of the page each.

`padding: 10px 10px 10px 10px;`

Sets the background color to tan.

`font-family: sans-serif;`

Creates some padding around the body of the page.

## *there are no* Dumb Questions

**Q:** CSS looks like a totally different language than HTML. Why have two languages? That's just more for me to learn, right?

**A:** You are quite right that HTML and CSS are completely different languages, but that is because they have very different jobs. Just like you wouldn't use English to balance your checkbook, or math to write a poem, you don't use CSS to create structure or HTML to create style because that's not what they were designed for. While this does mean you need to learn two languages,

you'll discover that because each language is good at what it does, this is actually easier than if you had to use one language to do both jobs.

**Q:** #d2b48c doesn't look like a color. How is #d2b48c the color "tan"?

**A:** There are a few different ways to specify colors with CSS. The most popular is called a "hex code," which is what #d2b48c is. This really is a tan color. For now, just go with it, and we'll be showing you exactly how #d2b48c is a color a little later.

**Q:** Why is there a "body" in front of the CSS rules? What does that mean?

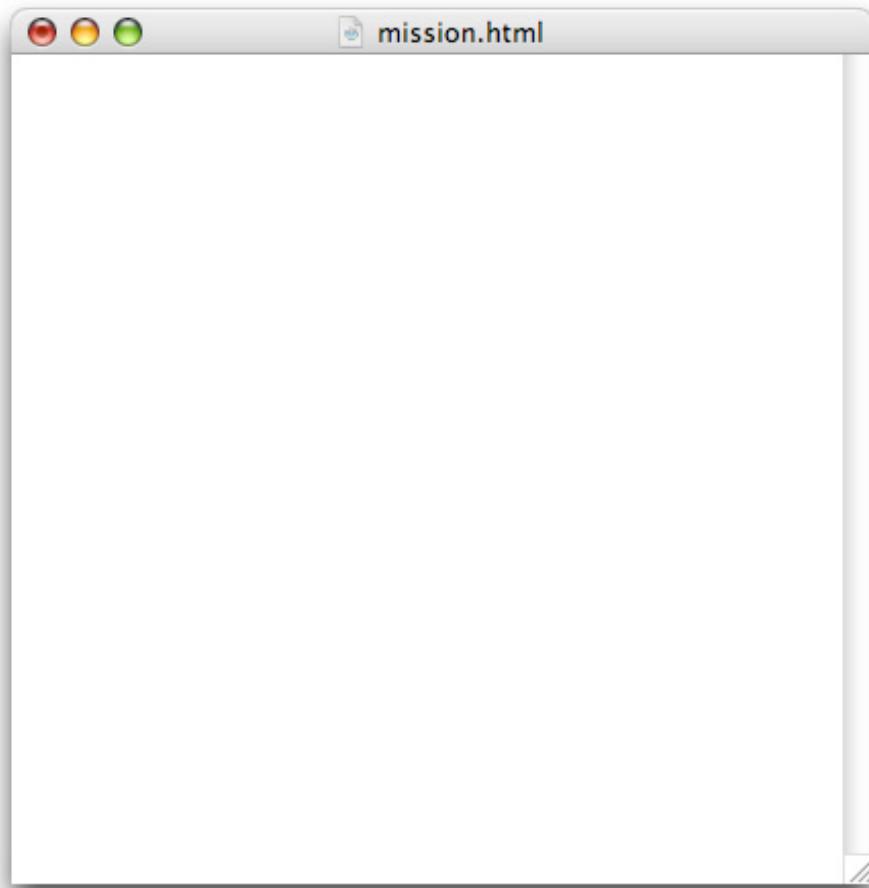
**A:** The "body" in the CSS means that all the CSS between the "{" and "}" applies to content within the HTML <body> element. So when you set the font to sans-serif, you're saying that the default font within the body of your page should be sans-serif.

We'll go into a lot more detail about how CSS works shortly, so keep reading. Soon, you'll see that you can be a lot more specific about how you apply these rules, and by doing so, you can create some pretty cool designs.



Now that you've put a little style in the Starbuzz "index.html" page, go ahead and update your "mission.html" page to have the same style.

- 1** Write the HTML for the "mission.html" page below, and then add the new CSS.
- 2** Update your "mission.html" file to include the new CSS.
- 3** Once you've done that, reload "mission.html" in your browser.
- 4** Make sure your mission page looks like ours at the end of the chapter.



## Fireside Chats



Tonight's talk: **HTML and CSS on content and style**

### **HTML**

Greetings, CSS; I'm glad you're here because I've been wanting to clear up some confusion about us.

Lots of people think that my tags tell the browsers how to *display* the content. It's just not true! I'm all about *structure*, not presentation.

Well, you can see how some people might get confused; after all, it's possible to use HTML without CSS and still get a decent-looking page.

Hey, I'm pretty powerful too. Having your content structured is much more important than having it look good. Style is so superficial; it's the structure of the content that matters.

Whoa, what an ego! Well, I guess I shouldn't expect anything else from you—you're just trying to make a fashion statement with all that style you keep talking about.

### **CSS**

Really? What kind of confusion?

Heck yeah—I don't want people giving you credit for my work!

"Decent" might be overstating it a bit, don't you think? I mean, the way most browsers display straight HTML looks kinda crappy. People need to learn how powerful CSS is and how easily I can give their web pages great style.

Get real! Without me, web pages would be pretty damn boring. Not only that, but take away the ability to style pages and no one is going to take your pages seriously. Everything is going to look clumsy and unprofessional.

## **HTML**

Right. In fact, we're totally different languages, which is good because I wouldn't want any of your style designers messing with my structure elements.

Yeah, that is obvious to me any time I look at CSS—talk about an alien language.

Millions of web writers would disagree with you. I've got a nice clean syntax that fits right in with the content.

Hey, ever heard of closing tags?

Just notice that no matter where you go, I've got you surrounded by <style> tags. Good luck escaping!

## **CSS**

Fashion statement? Good design and layout can have a huge effect on how readable and usable pages are. And you should be happy that my flexible style rules allow designers to do all kinds of interesting things with your elements without messing up your structure.

Don't worry, we're living in separate universes.

Yeah, like HTML can be called a language? Who has ever seen such a clunky thing with all those tags?

Just take a look at CSS; it's so elegant and simple, no goofy angle brackets <around> <everything>. <See> <I> <can><talk> <just><like><Mr.><HTML><,><look><at> <me><!>

Ha! I'll show you...because, guess what? I *can* escape...

↑  
Stay tuned!

Not only is this one fine cup of House Blend, but now we've got a web page to tell all our customers about our coffees. Excellent work. I've got some bigger ideas for the future; in the meantime, can you start thinking about how we are going to get these pages on the Internet so other people can see them?



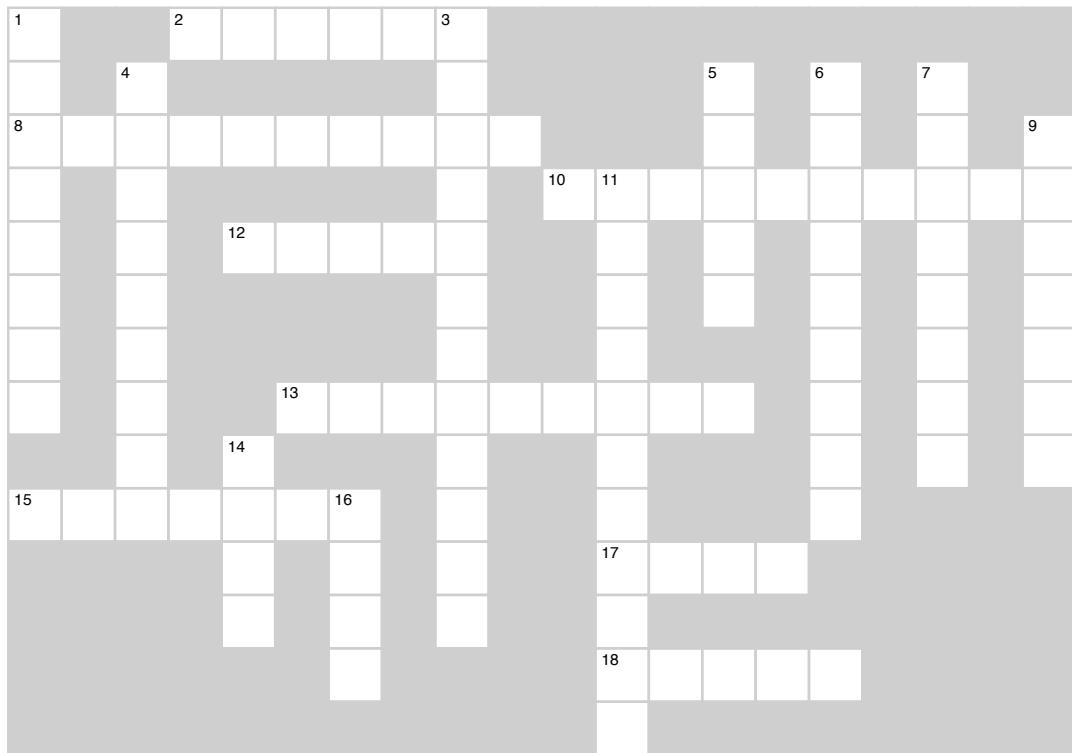
## BULLET POINTS

- HTML and CSS are the languages we use to create web pages.
- Web servers store and serve web pages, which are created from HTML and CSS. Browsers retrieve pages and render their content based on the HTML and CSS.
- HTML is an abbreviation for HyperText Markup Language and is used to structure your web page.
- CSS is an abbreviation for Cascading Style Sheets, and is used to control the presentation of your HTML.
- Using HTML, we mark up content with tags to provide structure. We call matching tags, and their enclosed content, elements.
- An element is composed of three parts: an opening tag, content, and a closing tag. There are a few elements, like `<img>`, that are an exception to this rule.
- Opening tags can have attributes. We've seen one already: type.
- Closing tags have a “/” after the left angle bracket, in front of the tag name, to distinguish them as closing tags.
- Your pages should always have an `<html>` element along with a `<head>` element and a `<body>` element.
- Information about the web page goes into the `<head>` element.
- What you put into the `<body>` element is what you see in the browser.
- Most whitespace (tabs, returns, spaces) is ignored by the browser, but you can use it to make your HTML more readable (to you).
- You can add CSS to an HTML web page by putting the CSS rules inside the `<style>` element. The `<style>` element should always be inside the `<head>` element.
- You specify the style characteristics of the elements in your HTML using CSS.



# HTMLcross

It's time to sit back and give your left brain something to do. It's your standard crossword; all of the solution words are from this chapter.



## Across

2. The "M" in HTML.
  8. Tags can have these to provide additional information.
  10. Browsers ignore this.
  12. You define presentation through this element.
  13. Purpose of <p> element.
  15. Two tags and content.
  17. What you see in your page.
  18. We emphasized Dance \_\_\_\_\_ Revolution.

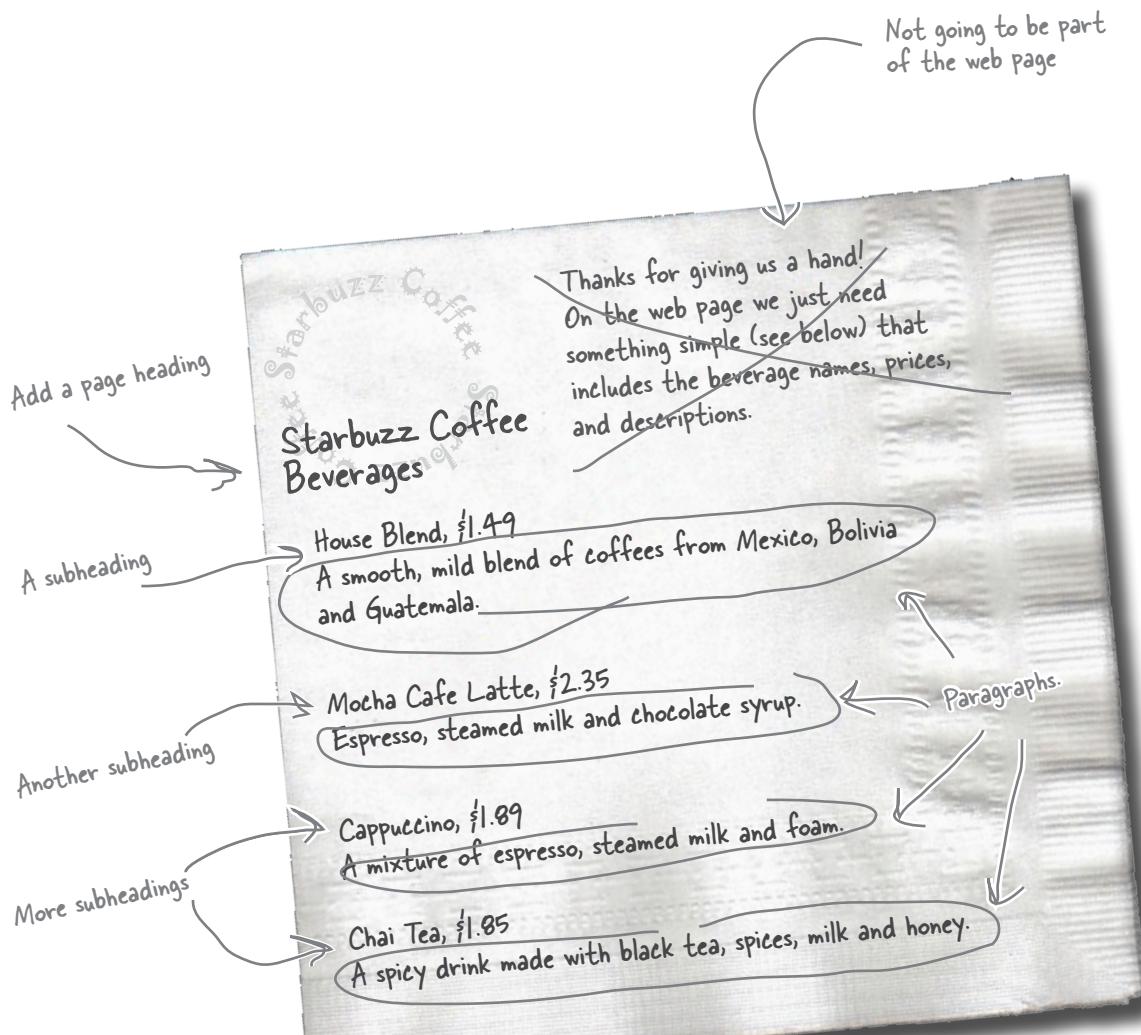
**Down**

1. There are six of these.
  3. CSS is used when you need to control this.
  4. You mark up content to provide this.
  5. Appears at the top of the browser for each page.
  6. Style we wish we could have had.
  7. Company that launched your web career.
  9. Only type of style available.
  11. Always separate these in HTML.
  14. About your web page.
  16. Opening and closing.



## Sharpen your pencil Solution

Go ahead and mark up the napkin (using your pencil) with any structure you see, and add anything that is missing.





## Markup Magnets Solution

Your job was to add some structure to the text from the Starbuzz napkin. Use the fridge magnets at the bottom of the page to mark up the text so that you've indicated which parts are headings, subheadings, and paragraph text. Here's our solution.

```

<h1> Starbuzz Coffee Beverages </h1>

<h2> House Blend, $1.49 </h2>
<p> A smooth, mild blend of coffees from Mexico, Bolivia and
Guatemala. </p>

<h2> Mocha Cafe Latte, $2.35 </h2>
<p> Espresso, steamed milk and chocolate syrup. </p>

<h2> Cappuccino, $1.89 </h2>
<p> A mixture of espresso, steamed milk and foam. </p>

<h2> Chai Tea, $1.85 </h2>
<p> A spicy drink made with black tea, spices, milk and honey. </p>

```

<h1>  
 </h1>  
 <h1>  
 </h1>

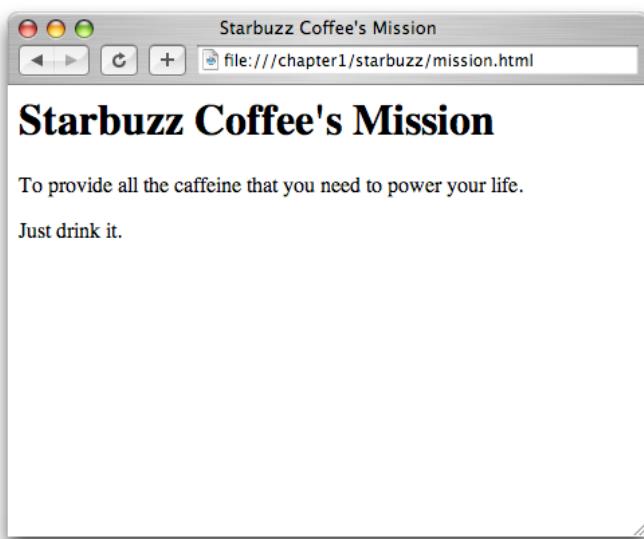


Leftover magnets



```
mission.html
```

```
<html>
  <head>
    <title>Starbuzz Coffee's Mission</title>
  </head>
  <body>
    <h1>Starbuzz Coffee's Mission</h1>
    <p>To provide all the caffeine that you need to power your life.</p>
    <p>Just drink it.</p>
  </body>
</html>
```



Here's the HTML.

Here's the HTML  
displayed in a browser.



## Exercise SOLUTION

Here's the CSS in the mission page.

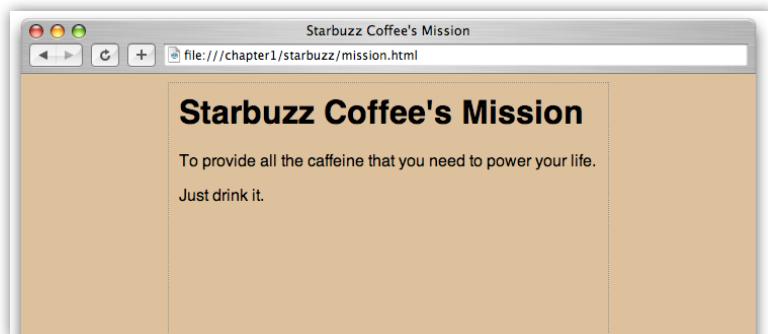


```

<html>
  <head>
    <title>Starbuzz Coffee's Mission</title>
    <style type="text/css">
      body {
        background-color: #d2b48c;
        margin-left: 20%;
        margin-right: 20%;
        border: 2px dotted black;
        padding: 10px 10px 10px 10px;
        font-family: sans-serif;
      }
    </style>
  </head>
  <body>
    <h1>Starbuzz Coffee's Mission</h1>
    <p>To provide all the caffeine that you need to power your life.</p>
    <p>Just drink it.</p>
  </body>
</html>

```

Now, the style matches the main Starbuzz page.





## WHO DOES WHAT?

Even though you've just glanced at CSS, you've already seen the beginnings of what it can do. Match each line in the style definition to what it does.

- |   |  |
|---|--|
| <code>background-color: #d2b48c;</code>                           | Defines the font to use for text.                                    |
| <code>margin-left: 20%;</code><br><code>margin-right: 20%;</code> | Defines a border around the body that is dotted and the color black. |
| <code>border: 2px dotted black;</code>                            | Sets the left and right margins to take up 20% of the page each.     |
| <code>padding: 10px 10px 10px 10px;</code>                        | Sets the background color to tan.                                    |
| <code>font-family: sans-serif;</code>                             | Creates some padding around the body of the page.                    |

'H		<sup>2</sup> M	A	R	K	U	<sup>3</sup> P						
E		<sup>4</sup> S			R			<sup>5</sup> T	<sup>6</sup> 5	<sup>7</sup> S			
<sup>8</sup> A	T	T	R	I	B	U	T	E	I	O	T	<sup>9</sup> T	
D	R				S		<sup>10</sup> W	<sup>11</sup> H	I	T	E	S	P
I	U		<sup>12</sup> S	T	Y	L	E	E	L	K	R	X	
N	C				N		A	E		I	B	T	
G	T				T		D			T	U	C	
S	U		<sup>13</sup> P	A	R	A	G	R	A	S	Z	S	
R		<sup>14</sup> H		T		N			C	Z	S		
<sup>15</sup> E	L	E	M	E	N	<sup>16</sup> T	I	D		H			
									<sup>17</sup> B	O	D	Y	
									O				
									<sup>18</sup> D	A	N	C	E
									Y				

# Want to read more?

You can [buy this book](#) at [oreilly.com](#)  
in print and ebook format.

**Buy 2 books, get the 3rd FREE!**

Use discount code: OPC10

All orders over \$29.95 qualify for **free shipping** within the US.

---

It's also available at your favorite book retailer,  
including the iBookstore, the [Android Marketplace](#),  
and [Amazon.com](#).



**O'REILLY®**

Spreading the knowledge of innovators

[oreilly.com](#)