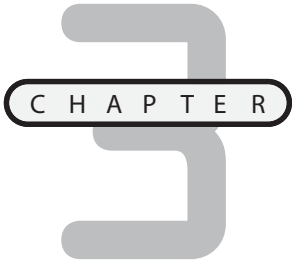
Part II

Working with (X)HTML



Creating (X)HTML Document Headings

All (X)HTML pages are made up of two primary sections: the head and body. This chapter focuses on the head section, which is responsible for specifying information about the documents and its content. With the exception of the `<title>` tag, which displays a text string in browser title bars, none of the contents of all of the head section are displayed. Instead, the tags stored in the head section are used to do things like provide information for search engines, define CSS style sheets used within documents, and define scripts embedded within the document.

Specifically, you will learn how to use:

- The `<title>` tag to display a text string in the browser's title bar
- The `<meta>` tag to provide keyword data used by search engines
- The `<base>` tag to reduce the size of links
- The `<style>` tag to create internal style sheets
- The `<link>` tag to link external style sheets to documents
- The `<script>` tag to add scripts to your documents

Project Preview: The Math Quiz Application

In this chapter, you will learn how to create a new web project call the Math Quiz application. To take the quiz, the user must have a pencil and a sheet of paper on which to write and record his answer. The Math Quiz presents the user with instructions for completing the quiz, as shown in Figure 3.1.

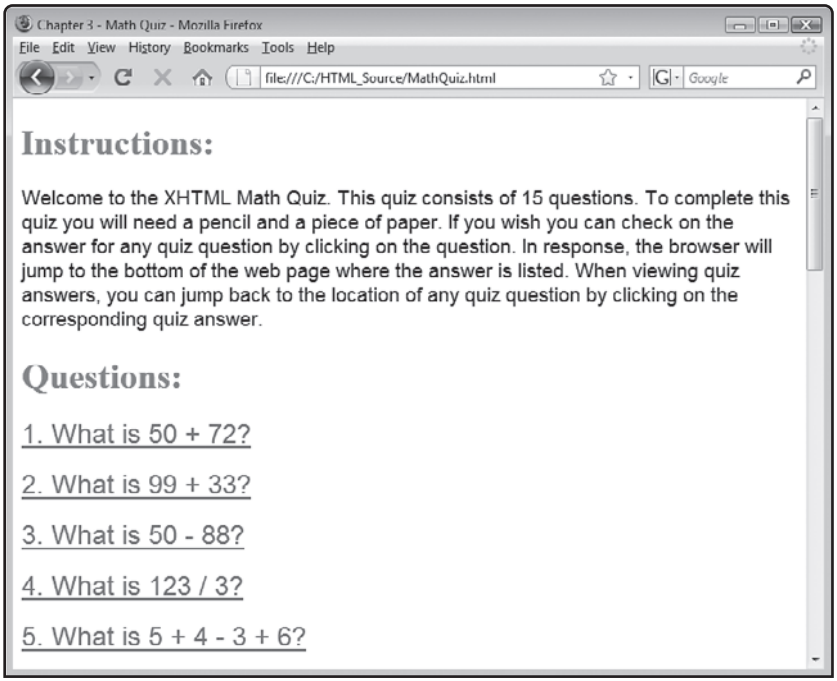


Figure 3.1
The quiz consists of 15 arithmetic questions.

As shown in Figure 3.2, the quiz consists of 15 separate math questions. Answers to all 15 questions are provided at the bottom of the web page.

Every question on the page is actually a link that when clicked jumps down and displays its corresponding answer. Likewise, every answer is also a link that when clicked returns you to its corresponding question. As this web page demonstrates, links can be used to control navigation within web pages and not just between them.

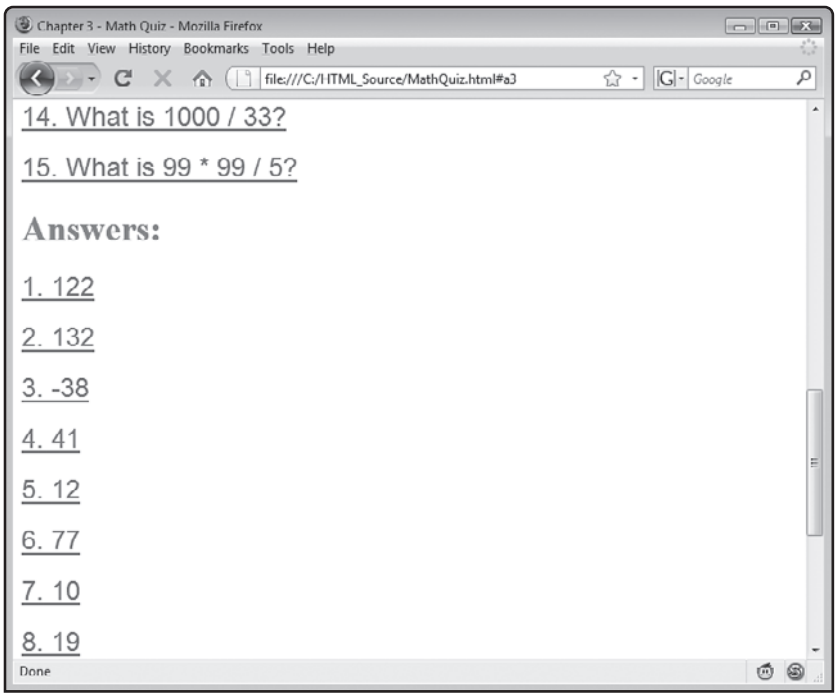


FIGURE 3.2
Answers to each question are provided at the bottom of the web page.

ESTABLISHING A DOCUMENT FRAMEWORK

Every (X)HTML document is made up of three parts, outlined here:

- `DOCTYPE` **declaration.** Tells the browsers what version of HTML or (X)HTML is being used.
- **Document head section.** Provides information about the document, including its title, style information, and scripts.
- **Document body.** Outlines the content that is rendered in the browser and made visible to visitors.

In Chapter 2 you learned how to specify a web document `DOCTYPE`. The focus of this chapter is on the development of a web document's head section. Chapters 4 to 6 will cover the development of a web document's `body` section.

Building a Document Template

While every (X)HTML page is unique, they all have these three parts in common. Going forward in this book, all examples will be presented using (X)HTML Strict. As such, it is a good idea to create a template that you can use as the basis for creating all other documents. As a first step in creating this template, create a document like the one shown here:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">

  <head>

  </head>

  <body>

  </body>
</html>
```

The `DOCTYPE` element is essential because it is responsible for telling the browser what version of (X)HTML should be applied when rendering the contents of the document. The specifics of the `DOCTYPE` statement were reviewed in Chapter 1. The rest of the document is contained within the `html` element, which itself is made up of the `body` and `head` elements.

To be well formed, the html element's opening <html> tag must at a minimum include an `xmlns` attribute, which is used to specify the XML namespace used by the (X)HTML document (`http://www.ww3.org/1999/xhtml`). Optionally, you may also include the `lang` and `xml:lang:` attributes. Both the `lang` and `xml:lang` attributes are used to specify the language in which the document has been written.

Making the Document Template Well Formed

If you take the document template that was just explained and validate it at `http://validator.w3.org`, you see that an error and two warnings are raised. The error and one of the warnings note that no character encoding is found. This can be corrected by adding a `<meta>` tag to the document's head section. The other warning occurs because a `<title>` tag has not been included in the head section. The following example addresses all of these problems and results in a well-formed (X)HTML page with no errors or warnings.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
```

```
<head>
  <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />
  <title></title>
</head>

<body>

</body>

</html>
```

Technically, the inclusion of the <meta> is not required. However, it is recommended that you use it anyway. This will help ensure that all browsers properly render your documents and will eliminate the warning messages displayed if you validate your web documents at validator.w3.org.

Adding Elements to the head Section

With the exception of the `<title>` tag, none of the tags that you place within the head section of your (X)HTML pages result in the display of any content that is visible to the user. However, the content that you add to the head section of your (X)HTML pages plays an important role in the presentation of documents and the interactivity of resulting web pages. For example, the head section is where you define a document's style rules, title, and the meta content that web search engines use when indexing your web pages.

The head section supports a number of different tags. The list of tags that you can use in the head section is outlined next. Their use is demonstrated in detail through the rest of this chapter.

- `<title>`—This tag displays a text string in the browser's title bar
- `<meta>`—This tag is used to provide keyword data used by Internet search engines
- `<base>`—This tag is used to shorten the size of links
- `<style>`—This tag is used to embed an internal style sheet into a document
- `<link>`—This tag is used to set up a link to an external style sheet document
- `<script>`—This tag is used to add scripts to your documents

Even though the contents of the head section do not get displayed by the web browser, visitors to your web pages can still see your web page's content by viewing its source. To do so, all users have to do is load your web page and then click on View > Source (Internet Explorer) or View > View Source (Apple Safari) or View > Page Source (Mozilla Firefox).

The <title> tag

The `<title>` tag is used to display a text string in the browser's title bar. You should always include a title on every document. The title clearly identifies the web page to visitors. Search engines display the title when they generate a list of web pages as the result of user queries. As a result, the title may be the first thing web surfers see when your site appears in a search engine's output.

Web browsers also use a document title when generating bookmarks for web pages and for the generation of bookmark names. When you formulate the text for your document, make sure that you keep it concise and yet descriptive of what the document is all about. The following document provides an example of how to use the `<title>`tag.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">

  <head>
    <title>XYZ's Graphic Design and Publishing</title>
  </head>

  <body>

  </body>

</html>
```

Note that all you have to do to supply a title for your document is to type the desired text between the `<title>` and `</title>` tags. Figure 3.3 demonstrates how the title appears when the document is loaded into Internet Explorer, Mozilla Firefox, and Apple Safari.

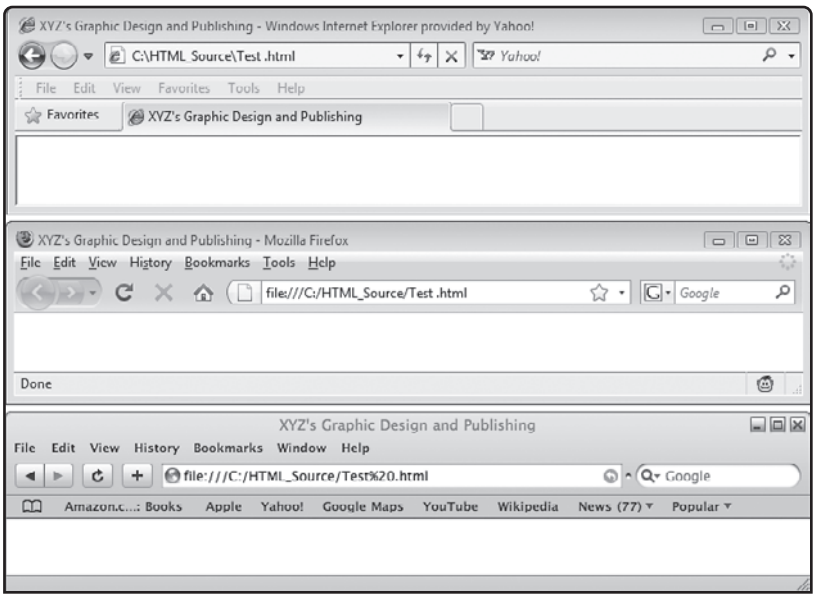


FIGURE 3.3
A demonstration of the use of the `title` element.

The <meta> tag

The most common means by which web surfers find new websites is through search engines. Search engines regularly search the Internet indexing new websites. You have the ability to influence the content that search engines collect through the use of metadata, documented in the head section of documents using `<meta>` tags. Metadata is a term used to describe data about data.

The use of metadata is an important component in the marketing of web pages. But it is only a small part of a well-constructed marketing campaign. To learn more about how to attract people to your website, check out *Increase Your Web Traffic in a Weekend* (ISBN# 1598634828) by Jerry Lee Ford, Jr.

Using meta tags, you can define suggested keywords and descriptions for search engines. For example, the following document contains three meta elements. The first element uses the `<meta>` tag's `name` element to specify a list of keywords for search engines to index. The words themselves are specified in a comma-separated list using the tag's `content` attribute. The second meta element specifies a descriptive statement for search engines to use their search result listing when displaying information about the web page. The third meta element, which you have seen before, uses the `http-equiv` attribute to connect or associate the value assigned to the `Content` attribute with the HTTP response header. The meta element also uses the `charset` attribute to specify the character-encoding scheme used by the document.

In addition to `keywords` and `description`, the `<meta>` tag's `name` attribute also supports `author` and `summary` as possible values.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">

  <head>
    <meta name="keywords" content="Games, Jokes, Riddles, Stories" />
    <meta name="description" content="Play free on-line games" />
    <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />
    <title>Chapter 1 - XHTML Joke</title>
  </head>

  <body>

  </body>

</html>
```

You can also use the meta tag's `http-equiv` attribute to instruct web browsers to perform certain actions. For example, the following element would be used to instruct the browser to automatically refresh the contents of the document every 30 seconds.

```
<meta http-equiv="refresh" content="30" />
```

Older web browsers may not support the use of the `<meta>` tag's `http- equiv` attribute. In which case, redirection will fail. As a result, it is a good idea to also display a link with a test message on your web page instructing visitors who are not automatically redirected to click on the link.

Similarly, you can use a meta element to automatically redirect visitors to another web page, as demonstrated here:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
```

```
  <head>
   <meta http-equiv="refresh" content="5; url= http://www.courseptr.com" />
   <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />
   <title>Redirection Example</title>
  </head>

  <body>
    <h1>The resource you are looking for has moved.</h1>
    <h2>Your browser will automatically be redirected in 5 seconds to:</h2>
    <p>
      <a href="http://www.courseptr.com">http://www.courseptr.com</a>
    </p>
  </body>

</html>
```

Here, visitors are redirected from the current web page to a page located at http://www.courseptr.com after a 5-second delay. In the event that the visitor's browser does not support meta tag redirection, a link has been provided for visitors to click on. Figure 3.4 shows an example of what visitors will see when this document is initially loaded into their browser.

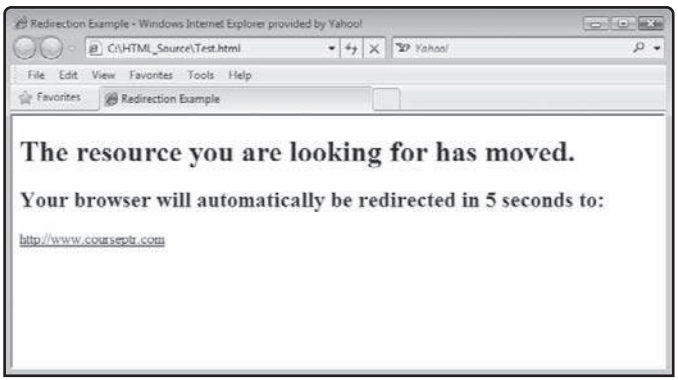


FIGURE 3.4
A simple redirection example.

The <base> tag

If you find that you are creating a web page that has a lot of links to files residing at the same location, you can shorten the URL reference to those files using the base element to specify the URL for all common links. The `<base>` tag has just one required attribute, `href`, which specifies the base URL for all of the links. By using the `<base>` tags to specify a base URL, you shorten the length of any URL that has the same base URL. This not only helps keep things simple but makes your documents easier to maintain. For example, if the location of all files

using the base URL changes, all you would have to do is modify the contents of the <base> tag to point to the new location and everything will work.

To see an example of how to set up a base URL, take a look at the following example.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">

  <head>
    <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />
    <base href="http://www.courseptr.com/" />
    <title>Demo - Working with the Base element</title>
  </head>

  <body>
    <h1>Book Categories</h1>
    <p><a href="ptr_catalog.cfm?group=Programming">Programming Topics</a>
      </p>
    <p><a href="ptr_catalog.cfm?group=Operating Systems">Operating
      Systems</a></p>
    <p><a href="ptr_catalog.cfm?group=Certification">Certification</a></p>
    <p><a href="ptr_catalog.cfm?group=Macintosh">Macintosh</a></p>
  </body>

</html>
```

When loaded, the browser will pre-append the base URL to each of the URLs in the four link elements. Figure 3.5 shows the resulting web page after it has been rendered by the browser.

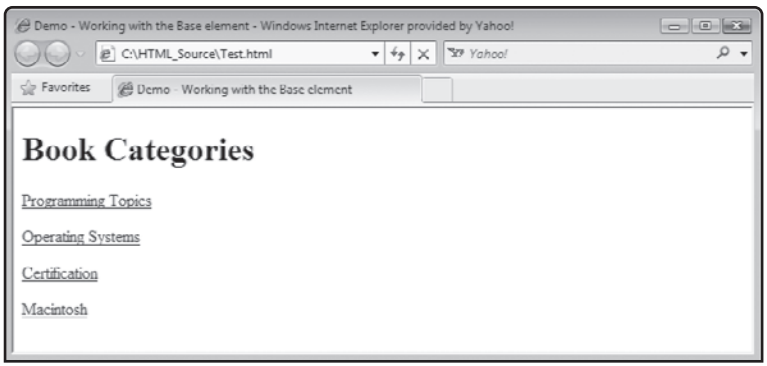


FIGURE 3.5

Using the base element to shorten the length of your links.

The <style> tag

There are several ways in which you can influence the way browsers present the appearance of your documents. These options include inline tags, internal style sheets, and external style sheets. Internal style sheets are embedded within web documents using the `<style>` tag. This tag has one required attribute named `type`, which must be assigned a value of `text/css`. The following web document demonstrates the use of the `<style>` tag.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">

  <head>
    <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />
    <style type="text/css">
      h1 {color: red;}
      a {font: 24px arial, courier;
         color: green;}
    </style>
    <title>Chapter 2 - LinkedJokes</title>
  </head>

  <body>
    <h1>Linked Jokes</h1>
    <a href="punchline1.html">What does Mickey Mouse's wife drive?</a><br />
    <a href="punchline2.html">How do you fix a tooth?</a>
  </body>

</html>
```

As you can see, the `style` element consists of both a start and an end tag. The starting style tag is written as `<style type="text/css">`. This internal style sheet consists of various rules that govern the presentation of the web document level 1 heading and link elements. Here, all level 1 headings are displayed in red and all links are displayed using a 24-pixel font. If the computer that has loaded the web document supports the arial font, that font is used. If the arial font is not present then the courier font is used. If that font is also not present, the computer default font is used. Figure 3.6 shows how this web document looks when loaded using Internet Explorer.

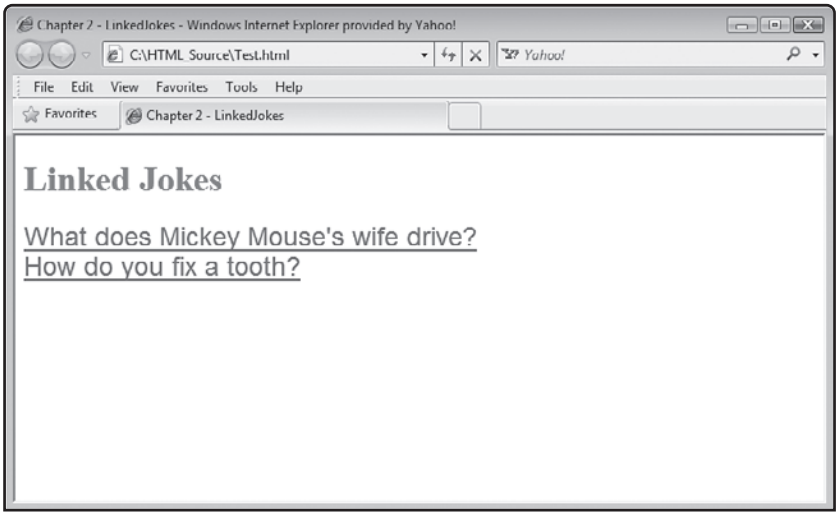


FIGURE 3.6

An example of a web page whose font type and color have been styled using an internal style sheet.

The previous web document contains two links. The first link points to a web page named punchline1.html and the second link points to a web page named punchline2.html. The contents of the punchline1.html file are shown next. Note that this file also contains its own internal style sheet. This style sheet contains a single rule that sets the font size, type, and color for all paragraph (p) elements.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">

  <head>
    <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />
    <style type="text/css">
      p {font: 24px arial, courier;
         color: green;}
    </style>
    <title>Chapter 2 - PunchLine1</title>
  </head>

  <body>
    <p>A Minnie-van</p>
 </body>

</html>
```

Figure 3.7 shows how the contents of this page appear when loaded by the web browser.

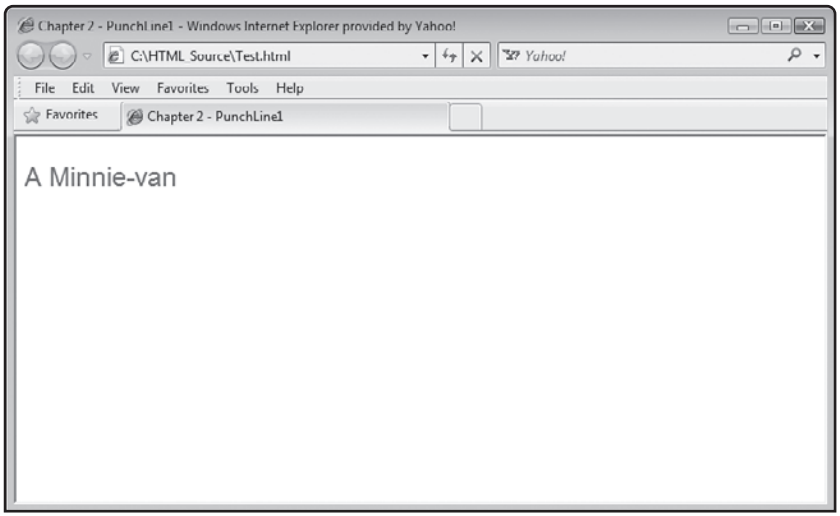


FIGURE 3.7

The text for the web page's paragraph (p) elements is displayed as green text.

The contents of the punchline2.html file are shown next. Note that this file also contains its own internal style sheet. This style sheet contains a single rule that sets the font size, type, and color for all paragraph (p) elements.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">

  <head>
    <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />
    <style type="text/css">
      p {font: 24px arial, courier;
         color: green;}
    </style>
    <title>Chapter 2 - PunchLine2</title>
  </head>

  <body>
    <p>With Toothpaste</p>
 </body>

</html>
```

Internal style sheets help to separate document structure from presentation. However, web developers seldom use internal style sheets, preferring instead to use external style sheets. You will learn all about internal and external style sheets in Chapters 7 and 8.

The <link> tag

The `link` element is used to define a relationship with another document. One common use of the `link` element is to establish a link to an external style sheet document. External style sheets, like their internal style sheet counterparts, provide web developers with the ability to style web page presentations. The following web document demonstrates how to use the `link` element to connect an external style sheet.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">

  <head>
    <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />
    <link rel="stylesheet" type="text/css" href="test.css" />
    <title>Chapter 2 - LinkedJokes</title>
  </head>

  <body>
    <h1>Linked Jokes</h1>
    <a href="punchline1.html">What does Mickey Mouse's wife drive?</a><br />
    <a href="punchline2.html">How do you fix a tooth?</a>
  </body>

</html>
```

As you can see, the `link` element consists of a single, self-closed tag. Note that in this example, the `link` element makes use of the following attributes.

- **`rel`.** Defines the relationship with the external document, which in the case of external style sheets is `stylesheet`.
- **`type`.** Specifies the MIME type of the linked document, which in the case of an external style sheet is `text/css`.
- **`href`.** Specifies the URL for the style sheet.

The name of the external style called in this example is test.css. This file is stored in the same folder as the web document. Like (X)HTML documents, CSS external style sheets are made up of plain text. The contents of this external style sheet are shown next.

```
h1 {color: red;}
a {font: 24px arial, courier; color: green;}
```

This external style sheet contains two rules. The first rules instruct the browser to display all headings in red and all links in green, using a 24-pixel font and either arial or courier font. As you can see, external style sheets do not include opening and closing `<style>` tags. Figure 3.8 shows an example of the contents of the external file sheet file.

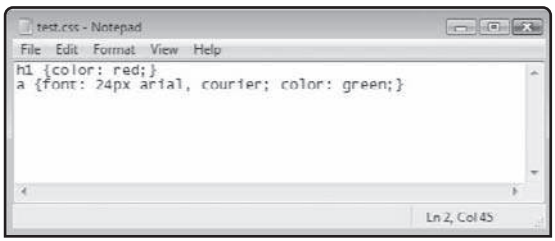


FIGURE 3.8
An example of a simple external style sheet.

The <script> tag

In order to create web pages that provide your visitors with an interactive experience, you need to learn how to work with one of any number of web-based computer-programming languages. One of the most popular and most widely used is JavaScript. *JavaScript* is a scripting language that executes within web browsers and which can be used to execute scripts embedded within web pages.

Although you can add scripts to the `head` or `body` sections of your documents, it is recommended that you always add them to the `head` section. This will ensure that they are loaded into memory before the browser renders the content in the `body` section, making the scripts available for use whenever they are needed. In order to add a script to your documents, you must use the `<script>` tag. The `<script>` tag supports a number of attributes, including:

- `language`. Identifies the script as JavaScript.
- `type`. Specifies the MIME type of the script.
- `scr`. Specifies a URL that provides a link to a file that contains a JavaScript.

The following web document demonstrates how to use the `script` element to embed a script within a web document.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

  <head>
    <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />
    <title>Chapter 1 - XHTML Joke</title>
    <script language="javascript" type="text/javascript">
    <!-- Start hiding JavaScript statements
      window.alert("Hello World!");
    //End hiding JavaScript statements -->
    </script>
  </head>

<body>

</body>

</html>
```

As you can see, since the JavaScript is embedded within the web document itself, only the `language` and `type` attributes were used. Figure 3.9 shows the output that is displayed when this script is loaded by the Apple Safari web browser.

FIGURE 3.9
Using JavaScript to display a message in a popup dialog window.

The following statements demonstrate how to call upon an external JavaScript from within a web document.

```
<script src="Test.js" language="javascript" type="text/javascript">
</script>
```

As you can see, the `src` attribute has been used to specify the location of the external JavaScript in place of embedded JavaScript statements.

Chapter 9 provides a good overview of JavaScript and its use in the dynamic generation of interactive content.

Back to the Math Quiz Page

Okay, it's time to turn your attention back to the development of the Math Quiz project. Once loaded into a web browser, this document will present your visitors with instructions for taking a 15-question math quiz along with the quiz itself. Answers to each question are provided at the bottom of the web page. Your visitors can jump back and forth between questions and answers by clicking on individual questions and answers. Specifically, every question is individually linked to its corresponding answer at the bottom of the page and every answer is linked back to its corresponding question.

As you can see, this chapter's project uses links to control navigation within a web page as opposed to using them to establish links to other external web pages, as was demonstrated in Chapter 2's project.

Designing the Application

As is the case with all of the web projects in this book, you will develop the Math Quiz project in a series of steps, as outlined here:

1. Create a new HTML document.
2. Develop the document's markup.
3. Test the HTML page.
4. Spice things up with a little CSS.
5. Perform a final test.

As you work on this web document, pay particular attention to how links are formed and note the appearance of the resulting web page content before and after the document's internal style sheet is applied.

Step 1: Creating a New HTML Document

The first step in creating this project's web document is to create an empty text file. Using your preferred code or text editor, create and save a new, empty text file. Name the file MathQuiz.html and save it in the same location as all of your other web documents.

Step 2: Developing the Document's Markup

The next step in creating the Math Quiz project is to write the web document's markup. Let's begin by adding the following standard set of elements to the MathQuiz.html document.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">

  <head>

  </head>

  <body>

  </body>

</html>
```

At this point you have supplied everything needed to create a valid, well-formed HTML page. All that remains is for you to add the markup required to complete the document's `head` and `body` section.

Updating the head Section

To complete the document's `head` section, modify it by embedding the `meta` and `title` elements, as shown here:

```
<head>

  <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />
  <title>Chapter 3 - Math Quiz</title>

</head>
```

The `meta` element specifies the document's content type and character set, and the `title` element specifies a text string to be displayed in the web browser's titlebar.

Specifying Document Content

The web document content is provided by the document's `body` section, which is made up of a number or heading (`h1`), paragraph (`p`), and link (`a`) elements. Begin the development of the `body` section's markup by adding the following statements to it.

```
<h1>Instructions:</h1>

<p>Welcome to the XHTML Math Quiz. This quiz consists of 15 questions.
   To complete this quiz you will need a pencil and a piece of paper.
   If you wish you can check on the answer for any quiz question by
   clicking on the question. In response, the browser will jump to the
   bottom of the web page where the answer is listed. When viewing quiz
   answers, you can jump back to the location of any quiz question
   by clicking on the corresponding quiz answer.</p>
```

As you can see, the `h1` element encloses a heading that identifies the location of the quiz's instructions, which are embedded within a `p` element that follows the `h1` element. Next, add the following statement to the end of the `body` section. These statements consist of another `h1` element that identifies the beginning of the quiz's list of questions followed by a series of 15 `p` elements. Embedded within each paragraph is a link (`a`) element. Each link is identified by a unique `id` assignment. Each link `href` attribute points to a named element located at the end of the page. The text for each link is then displayed.

One of the objectives of this book is to provide you with a new web project at the end of each chapter. Unfortunately, the tradeoff of this approach means that in the early chapters you must use a number of (X)HTML elements before they have been formally instructed and explained. All of the elements that you see in this project are explained in detail in Chapter 4. For now, just follow along with the explanations that are provided and type in everything exactly as you see it. If necessary, you can return and review this project again after completing Chapter 4.

```
<h1>Questions:</h1>

<p><a id="q1" href="#a1">1. What is 50 + 72?</a></p>
<p><a id="q2" href="#a2">2. What is 99 + 33?</a></p>
<p><a id="q3" href="#a3">3. What is 50 - 88?</a></p>
<p><a id="q4" href="#a4">4. What is 123 / 3?</a></p>
<p><a id="q5" href="#a5">5. What is 5 + 4 - 3 + 6?</a></p>
<p><a id="q6" href="#a6">6. What is 7 * 11?</a></p>
<p><a id="q7" href="#a7">7. What is 4 * 5 / 2</a></p>
<p><a id="q8" href="#a8">8. What is 10 * 2 + 6 - 7?</a></p>
<p><a id="q9" href="#a9">9. What is 50 + 72?</a></p>
<p><a id="q10" href="#a10">10. What is 50 / 5 * 6?</a></p>
```

```
<p><a id="q11" href="#a11">11. What is 3 * 3 * 3 * 3 / 9?</a></p>
<p><a id="q12" href="#a12">12. What is 5 / 2 + 4.5 + 3?</a></p>
<p><a id="q13" href="#a13">13. What is 22 - 5 - 3 - 1?</a></p>
<p><a id="q14" href="#a14">14. What is 1000 / 33?</a></p>
<p><a id="q15" href="#a15">15. What is 99 * 99 / 5?</a></p>
```

Now that we have written all of the markup required to outline the web document's questions, it is time to add the markup responsible for displaying answers to all of the quiz's questions. As was the case in the markup used to display the quiz questions, the markup for the answers also consists of `a` (link) elements embedded within `p` (paragraph) elements. The markup is provided below and should be added to the end of the document's `body` section.

```
<h1>Answers:</h1>

<p><a id="a1" href="#q1">1. 122</a></p>
<p><a id="a2" href="#q2">2. 132</a></p>
<p><a id="a3" href="#q3">3. -38</a></p>
<p><a id="a4" href="#q4">4. 41</a></p>
<p><a id="a5" href="#q5">5. 12</a></p>
<p><a id="a6" href="#q6">6. 77</a></p>
<p><a id="a7" href="#q7">7. 10</a></p>
<p><a id="a8" href="#q8">8. 19</a></p>
<p><a id="a9" href="#q9">9. 122</a></p>
<p><a id="a10" href="#q10">10. 60</a></p>
<p><a id="a11" href="#q11">11. 9</a></p>
<p><a id="a12" href="#q12">12. 10</a></p>
<p><a id="a13" href="#q13">13. 15</a></p>
<p><a id="a14" href="#q14">14. 30.30</a></p>
<p><a id="a15" href="#q15">15. 1960.2</a></p>
```

Step 3: Performing a Quick Test of the Document

At this point, your copy of the Math Quiz application should be ready for testing. To do so, open your preferred web browser and then execute the following procedure.

1. Click on the browser's File menu and select the Open File command. This displays a standard file open dialog.
2. Using the dialog window, navigate to the folder where you stored the web page and select it.
3. Click on the Open button. The browser will then load and display the specified document.

Once loaded and rendered in your browser, the Math Quiz should look like the example shown in Figure 3.10.

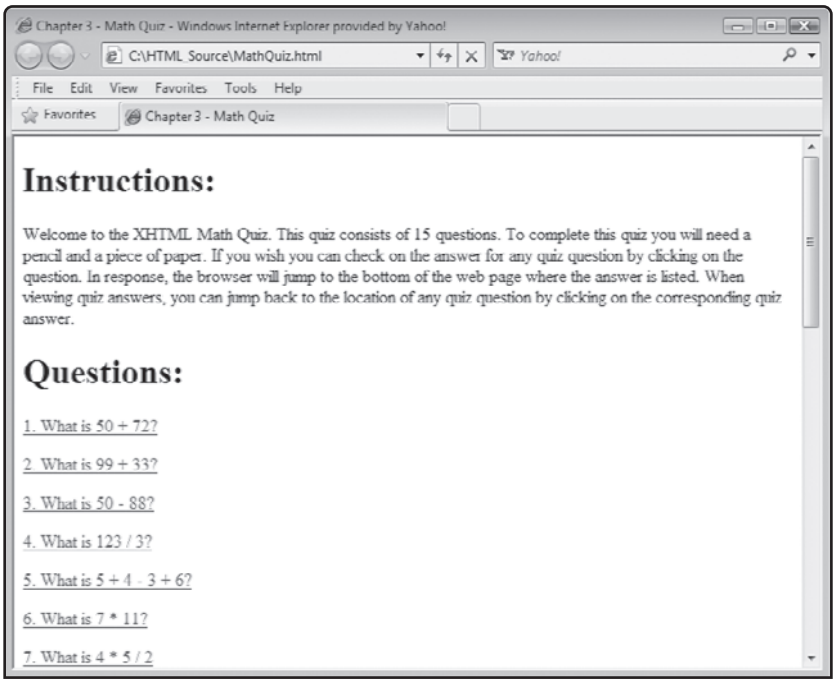


Figure 3.10
The quiz consists of arithmetic questions.

As you can see, by default, the text displayed on the resulting web page is displayed in black, except for the question and answer links, which are displayed in blue. The page's content is displayed using the browser's default font type and size.

Step 4: Spicing Things Up with an Internal Style Sheet

Rather than accept this default presentation, let's spice things up a bit by adding a CSS internal style sheet to the web document. Specifically, you will modify the presentation of the document by modifying its default colors, font sizes, and font types.

Embedding an Internal Style Sheet

As you learned earlier in this chapter, to add an internal style sheet to a web document, you must add the `style` element to the document's `head` section. To do so, embed the following CSS markup to the document's `head` section.

```
<style type="text/css">
  h1 {color: red;}
  a {font: 24px arial, courier;
     color: green;}
  p {color: black;
     font: 18px arial, courier;}
</style>
```

(X)HTML uses elements made up of tags to organize content. (X)HTML tags are created by placing them within < and > characters. CSS, on the other hand, uses rules when specifying presentation. CSS rules are embedded within { and } characters. The first rule listed above instructs the browser to display all h1 (heading) elements using the color red. The next CSS rule instructs the browser to display the document's links using a font that is 24 pixels high using arial font. If the arial font is not supported on the user's computer, courier is used. Lastly, link text is displayed in green color.

The third and final CSS rule instructs the browser to display all p (paragraph) text in black, using a font size of 18 pixels and the arial font (courier is not supported).

A detailed review of CSS and its syntax is provided in Chapters 7 and 8.

The Finished HTML Document

Your copy of the Math Quiz's web document should now be complete. To make sure that you have assembled it correctly, compare your document against the following completed example.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">

  <head>

    <meta http-equiv="Content-type" content="text/xhtml; charset=UTF-8" />

    <style type="text/css">
      h1 {color: red;}
      a {font: 24px arial, courier;
```

```
      color: green;}
    p {color: black;
      font: 18px arial, courier;}
  </style>

  <title>Chapter 3 - Math Quiz</title>

</head>

<body>

  <h1>Instructions:</h1>

  <p>Welcome to the XHTML Math Quiz. This quiz consists of 15 questions.
     To complete this quiz you will need a pencil and a piece of paper.
     If you wish you can check on the answer for any quiz question by
     clicking on the question. In response, the browser will jump to the
     bottom of the web page where the answer is listed. When viewing quiz
     answers, you can jump back to the location of any quiz question
     by clicking on the corresponding quiz answer.</p>

  <h1>Questions:</h1>

  <p><a id="q1" href="#a1">1. What is 50 + 72?</a></p>
  <p><a id="q2" href="#a2">2. What is 99 + 33?</a></p>
  <p><a id="q3" href="#a3">3. What is 50 - 88?</a></p>
  <p><a id="q4" href="#a4">4. What is 123 / 3?</a></p>
  <p><a id="q5" href="#a5">5. What is 5 + 4 - 3 + 6?</a></p>
  <p><a id="q6" href="#a6">6. What is 7 * 11?</a></p>
  <p><a id="q7" href="#a7">7. What is 4 * 5 / 2</a></p>
  <p><a id="q8" href="#a8">8. What is 10 * 2 + 6 - 7?</a></p>
  <p><a id="q9" href="#a9">9. What is 50 + 72?</a></p>
  <p><a id="q10" href="#a10">10. What is 50 / 5 * 6?</a></p>
  <p><a id="q11" href="#a11">11. What is 3 * 3 * 3 * 3 / 9?</a></p>
  <p><a id="q12" href="#a12">12. What is 5 / 2 + 4.5 + 3?</a></p>
  <p><a id="q13" href="#a13">13. What is 22 - 5 - 3 - 1?</a></p>
  <p><a id="q14" href="#a14">14. What is 1000 / 33?</a></p>
  <p><a id="q15" href="#a15">15. What is 99 * 99 / 5?</a></p>
```

```
    <h1>Answers:</h1>

    <p><a id="a1" href="#q1">1. 122</a></p>
    <p><a id="a2" href="#q2">2. 132</a></p>
    <p><a id="a3" href="#q3">3. -38</a></p>
    <p><a id="a4" href="#q4">4. 41</a></p>
    <p><a id="a5" href="#q5">5. 12</a></p>
    <p><a id="a6" href="#q6">6. 77</a></p>
    <p><a id="a7" href="#q7">7. 10</a></p>
    <p><a id="a8" href="#q8">8. 19</a></p>
    <p><a id="a9" href="#q9">9. 122</a></p>
    <p><a id="a10" href="#q10">10. 60</a></p>
    <p><a id="a11" href="#q11">11. 9</a></p>
    <p><a id="a12" href="#q12">12. 10</a></p>
    <p><a id="a13" href="#q13">13. 15</a></p>
    <p><a id="a14" href="#q14">14. 30.30</a></p>
    <p><a id="a15" href="#q15">15. 1960.2</a></p>

  </body>

</html>
```

Step 5: Loading and Testing the Math Quiz

Okay, it is time to retest your copy of the Math Quiz document to ensure that it looks and works as previously described. If your web browser is still open from the last time you tested the web page, just click on its relation\refresh button. Otherwise, open your browser again and reload your document. Figure 3.11 shows the resulting web page that is rendered when the document is loaded.

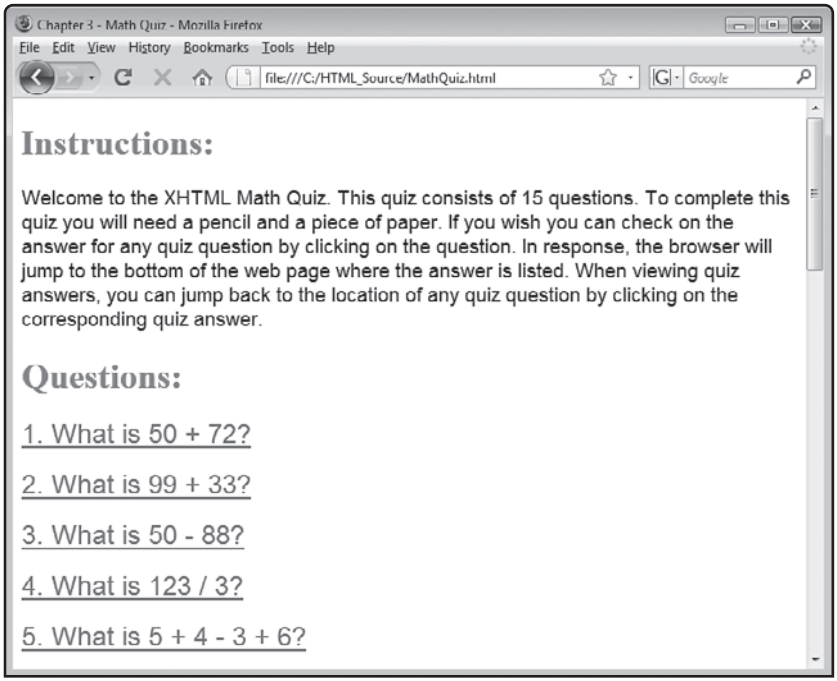


FIGURE 3.11
The quiz consists of 15 arithmetic questions.

SUMMARY

This chapter focused on the development of your (X)HTML page's `head` section. This included learning how to use the `<title>` tag to display a text string in the browser's title bar and the `<meta>` tag to provide keyword and descriptive data for search engines. You also learned how to use the `<base>` tags to reduce tag size and the `<style>` and `<link>` tags to work with internal and external style sheets. Lastly, you learned how to work with the `<script>` tag to add scripts to your documents. On top of all this, you learned how to create the Math Quiz web application.

Challenges

1. Consider expanding on the Math Quiz by adding additional questions and answers to it.
2. Improve web page navigation by adding an extra link at the top and bottom of the document, which when clicked jumps the user from the top to the bottom of the web page and vice versa.
3. Consider expanding the text that provides the user with instructions, explaining the number of questions that must be answered in order to pass the quiz.
4. Try to modify the web page's presentation by experimenting with the rules located in its internal style sheet, assigning different font colors, font types, and sizes.