HTML5 Application Development Fundamentals, Exam 98-375

VP & PUBLISHER Don Fowley
EDITOR Bryan Gambrel **DIRECTOR OF SALES** Mitchell Beaton **EXECUTIVE MARKETING MANAGER** Chris Ruel
MICROSOFT PRODUCT MANAGER Rob Linsky of Microsoft Learning **EDITORIAL PROGRAM ASSISTANT** Jennifer Lartz
ASSISTANT MARKETING MANAGER Debbie Martin **SENIOR PRODUCTION MANAGER** Janis Soo
ASSOCIATE PRODUCTION MANAGER Joel Balbin
CREATIVE DIRECTOR Harry Nolan
COVER DESIGNER Georgina Smith **TECHNOLOGY AND MEDIA** Tom Kulesa/Wendy Ashenberg

This book was set in Garamond by Aptara, Inc. and printed and bound by Bind-Rite Robbinsville.
The cover was printed by Bind-Rite Robbinsville.

The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

The book expresses the author's views and opinions. The information contained in this book is provided without any express, statutory, or implied warranties. Neither the authors, John Wiley & Sons, Inc., Microsoft Corporation, nor their resellers or distributors will be held liable for any damages caused or alleged to be caused either directly or indirectly by this book.

ISBN 978-1-118-35993-8

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

www.wiley.com/college/microsoft *or*
call the MOAC Toll-Free Number: 1+(888) 764-7001 (U.S. & Canada only)

Wiley's publishing vision for the Microsoft Official Academic Course series is to provide students and instructors with the skills and knowledge they need to use Microsoft technol ogy effectively in all aspects of their personal and professional lives. Quality instruction is required to help both educators and students get the most from Microsoft's software tools and to become more productive. Thus our mission is to make our instructional programs trusted educational companions for life.

To accomplish this mission, Wiley and Microsoft have partnered to develop the highest quality educational programs for Information Workers, IT Professionals, and Developers. Materials cre ated by this partnership carry the brand name "Microsoft Official Academic Course," assuring
instructors and students alike that the content of these textbooks is fully endorsed by Microsoft, and that they provide the highest quality information and instruction on Microsoft products. The Microsoft Official Academic Course textbooks are "Official" in still one more way—they are the officially sanctioned courseware for Microsoft IT Academy members.

The Microsoft Official Academic Course series focuses on *workforce development*. These programs are aimed at those students seeking to enter the workforce, change jobs, or embark on new careers as information workers, IT professionals, and developers. Microsoft Official Academic Course programs address their needs by emphasizing authentic workplace scenarios with an abundance of projects, exercises, cases, and assessments.

The Microsoft Official Academic Courses are mapped to Microsoft's extensive research and job-task analysis, the same research and analysis used to create the Microsoft Technology Associate (MTA) and Microsoft Certified Solutions Developer (MCSD) exams. The text books focus on real skills for real jobs. As students work through the projects and exercises in the textbooks they enhance their level of knowledge and their ability to apply the latest Microsoft technology to everyday tasks. These students also gain resume-building cre dentials that can assist them in finding a job, keeping their current job, or in furthering their education.

The concept of life-long learning is today an utmost necessity. Job roles, and even whole job categories, are changing so quickly that none of us can stay competitive and productive without continuously updating our skills and capabilities. The Microsoft Official Academic Course offerings, and their focus on Microsoft certification exam preparation, provide a means for people to acquire and effectively update their skills and knowledge. Wiley sup ports students in this endeavor through the development and distribution of these courses as

Microsoft's official academic publisher.

Today educational publishing requires attention to providing quality print and robust elec tronic content. By integrating Microsoft Official Academic Course products, *WileyPLUS*, and Microsoft certifications, we are better able to deliver efficient learning solutions for students and teachers alike.

Joseph Heider

General Manager and Senior Vice President

Welcome to the Microsoft Official Academic Course (MOAC) program for HTML5 Application Development Fundamentals. MOAC represents the collaboration between Microsoft Learning and John Wiley & Sons, Inc. publishing company. Microsoft and Wiley teamed up to produce a series of textbooks that deliver compelling and innovative teaching solutions to instructors and superior learning experiences for students. Infused and informed by in-depth knowledge from the creators of Microsoft products, and crafted by a publisher known worldwide for the pedagogical quality of its products, these textbooks maximize skills transfer in minimum time. Students are challenged to reach their potential by using their new technical skills as highly productive members of the workforce.

Because this knowledge base comes directly from Microsoft, creator of the Microsoft Certified Solutions Developer (MCSD) and Microsoft Technology Associate (MTA) exams (www.microsoft.com/learning/certification), you are sure to receive the topical coverage that is most relevant to students' personal and professional success. Microsoft's direct participation not only assures you that MOAC textbook content is accurate and current; it also means that students will receive the best instruction possible to enable their success on certification exams and in the workplace.

▪ The Microsoft Official Academic Course Program

The *Microsoft Official Academic Course* series is a complete program for instructors and institutions to prepare and deliver great courses on Microsoft software technologies. With MOAC, we recognize that, because of the rapid pace of change in the technology and curriculum developed by Microsoft, there is an ongoing set of needs beyond classroom instruction tools for an instructor to be ready to teach the course. The MOAC program endeavors to provide solutions for all these needs in a sys

tematic manner in order to ensure a successful and rewarding course experience for both instructor and student—technical and curriculum training for instructor readiness with new software releases; the software itself for student use at home for building hands-on skills, assessment, and validation of skill development; and a great set of tools for delivering instruction in the classroom and lab. All are important to the smooth delivery of an interesting course on Microsoft software, and all are pro vided with the MOAC program. We think about the model below as a gauge for ensuring that we completely support you in your goal of teaching a great course. As you evaluate your instructional

materials options, you may wish to use the model for comparison purposes with available

INSTRUCTOR READINESS
STUDENT SOFTWARE
LEARNING VALIDATION
STUDENT ASSESSMENT
CLASSROOM TOOLS

products.

▪ Pedagogical Features Pedagogical Features

The MOAC textbook for HTML5 Application Development Fundamentals is designed to cover all the learning objectives for that MTA exam 98-375, which is referred to as its "objective domain." The Microsoft Technology Associate (MTA) exam objectives are high lighted throughout the textbook. Many pedagogical features have been developed specifically for *Microsoft Official Academic Course* programs.

Presenting the extensive procedural information and technical concepts woven throughout the textbook raises challenges for the student and instructor alike. The Illustrated Book Tour that follows provides a guide to the rich features contributing to *Microsoft Official Academic Course* program's pedagogical plan. Following is a list of key features in each lesson designed to prepare students for success as they continue in their IT education, on the certification exams, and in the workplace:

- Each lesson begins with an **Exam Objective Matrix**. More than a standard list of learning objectives, the Exam Objective Matrix correlates each software skill covered in the lesson to the specific exam objective domain.
- Concise and frequent **Step-by-Step** instructions teach students new features and provide an opportunity for hands-on practice. Numbered steps give detailed, step-by-step instructions to help students learn software skills.
- **Illustrations:** Screen images provide visual feedback as students work through the exercises. The images reinforce key concepts, provide visual clues about the steps, and allow students to check their progress.
- **Key Terms:** Important technical vocabulary is listed with definitions at the beginning of the lesson. When these terms are used later in the lesson, they appear in bold italic type and are defined. The Glossary contains all of the key terms and their definitions.

- Engaging point-of-use **Reader Aids**, located throughout the lessons, tell students why this topic is relevant (*The Bottom Line*) and provide students with helpful hints (*Take Note*). Reader Aids also provide additional relevant or background information that adds value to the lesson.
- **Certification Ready** features throughout the text signal to students where a specific certification objective is covered. They provide students with a chance to check their understanding of that particular MTA objective and, if necessary, review the section of the lesson where it is covered. MOAC offers complete preparation for MTA certification.
- **End-of-Lesson Questions:** The Knowledge Assessment section provides a variety of multiple-choice, true-false, matching, and fill-in-the-blank questions.
- **End-of-Lesson Exercises:** Competency Assessment case scenarios and Proficiency Assessment case scenarios are projects that test students' ability to apply what they've learned in the lesson.

Lesson Features

Key Terms

KEY TERMS

Building the User

LESSON

Interface by Using HTML5: Organization, Input, and Validation

Exam Objective Matrix

EXAM OBJECTIVE MATRIX

S /CONCEPTS MTA EXAM OBJECTIVE MTA EXAM OBJECTIVE N

Choosing and Configuring Choose and configure 2.4
HTML5 Tags to Organize HTML5 tags to organize
Content and Forms content and forms.

Choosing and Configuring HTML5 Choose and configure HTML5 2.5
Tags for Input and Validation tags for input and validation.

article element **aside** element
autofocus attribute automatic validation client-side validation **datalist**
element **email** attribute
footer element
form input
global attribute **header** element **menu** element
nav element
ordered list
pattern attribute
placeholder text
required attribute
section element
semantic markup
server-side validation
table
unordered list
validation
Web form

One of your new tasks as an intern at Malted Milk Media is to create a Web form that restricts what a user can enter into the form fields and validates the input. To prepare to create the Web form, you must first learn how best to organize or structure the markup using new HTML5 elements.

53

Business Scenario

c01ManagingtheApplicationLifeCycle.indd Page 5 10/16/12 3:36 PM user-f494
/Users/F-440/Desktop

Managing the Application Life Cycle | 5

More Information Reader Aid

MORE INFORMATION

The Web addresses used in the previous exercises are repeated here for your convenience—consider
bookmarking them for later use. You can find information on the HTML5 standard on the W3C Web site at
http://www.w3.org/TR/html5/. One Web site that provides HTML5 tutorials is at http://www.w3schools.com/
html5/default.asp. Microsoft provides useful information to new developers at the Beginner Developer Learning
Center (http://msdn.microsoft.com/en-us/beginner/default.aspx) and the MSDN site at http://bit.ly/Hd9uzt.

Creating Apps

HTML5 apps are much easier to develop than similar apps that use other technologies
such as Adobe Flash and Microsoft Silverlight. HTML5, CSS, and JavaScript are interpreted languages, which means they do not require compilation. You can debug the
code within a browser, make quick edits, and then refresh the browser window to see the
result of the changes.

Although you can create a lot of code using a simple text or HTML editor, if you want to
package and deploy your app, you'll need to use an application development tool such as
Microsoft Visual Studio.

Figure 1-3

General steps for creating an app
Plan

Design UI Build App Validate

Deploy

Test and Debug Package

Update Manifest Write Code

You are the new intern at Malted Milk Media, a creative agency that creates rich media applications for clients. The company will soon begin using HTML5 in its projects. Your manager asked you to research HTML5 and its related technologies and present a report on significant changes from HTML 4.01 to HTML5. You must also include information on how HTML5 can be used for creating touch-screen applications, such as those for PCs, slates, tablets, and smartphones.

■

HTML5 is the latest HTML standard and a family of technologies that includes HTML, CSS, and JavaScript. Although the HTML5 standard won't be

THE BOTTOM LINE

*

W3C® is a trademark (registered in numerous countries) of the World Wide Web Consortium; marks of W3C are reg istered and held by its host institutions MIT, ERCIM, and Keio.

Figure 1-1

The HTML5 logo

*

Even though HTML5 is still under develop ment, most of the major Web browsers such
as Microsoft Internet Explorer, Google Chrome, Mozilla Firefox, Opera, and Apple Safari support many HTML5 elements.

1.1

most modern Web browsers already support HTML5 elements, and HTML5 app development for Web and mobile device browsers is well underway.

Hypertext Markup Language (HTML) is the language you use to describe Web pages. It is a ***markup language***, not a programming language, which means HTML uses markup tags such as `<body>` and `<h1>` to describe parts of a Web page. An HTML file doesn't "run" like a program. Instead, an HTML file is interpreted by a browser to display a Web page based on the tags.

Since 1999, HTML 4.01 has been the standard for Web pages, but the world has changed quite a bit since then. Web users want richer Web applications that incorporate audio, video, and a lot of interactivity on the Web sites they visit. And with the surge in popularity of mobile devices like slates, tablets, and smartphones, users want to experience the same rich ness and interactivity in mobile applications regardless of which device they choose.

finalized for a few years,
In this section, you will learn the general steps involved in creating an app. Figure 1-3 illustrates the steps, which are explained as follows:

- **Plan your project:** Think about the type of app you want to create. Whatever you choose, it's best to keep it simple while you're first learning HTML5 app development. After you decide on the main action of your app, create an outline of the general flow of the application from start to finish. Also determine the type of user interactivity you want to include, such as a touch interface, whether you need to save data outside of the app, and whether the app should connect to other apps or services (such as an RSS feed).
- **Design a UI:** When designing the user interface, determine how you want the app to appear to users. Group the content in a way that makes logical sense. Begin listing the commands you will need for the app to run as expected, and gather images and multimedia clips if necessary. At a minimum, today's apps require a ***launcher icon***,

All of this has prompted the need for a new standard, which will be ***HTML5***. The ***World Wide Web Consortium (W3C)*** is the main standards body developing specifications for HTML5, which should be finalized in 2014. The HTML5 logo is shown in Figure 1-1.

This work is attributed to the W3C.

An important point to remember about HTML5 is that it is both a standard and a combination or family of new HTML markup tags, CSS, JavaScript, and other related technologies. ***Cascading Style Sheets (CSS)*** defines styles for HTML in a separate file, so you can easily change fonts, font sizes, and other attributes in a CSS file and the changes are reflected across all HTML files that ref erence the CSS file. The latest version of CSS is CSS3. ***JavaScript*** is a ***scripting language*** (a pro gramming language that uses scripts and requires no compiler) that adds interactivity to Web pages.

Although you can use HTML5, CSS3, and JavaScript to create Web pages, you can also use the combination to develop client applications (apps) that run on touch-enabled devices like

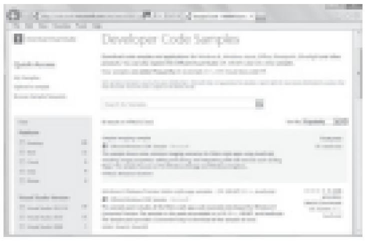
Bottom Line

which represents your app.

- **Update the app manifest:** Every app requires a manifest file. The manifest file describes properties of the app and what the app needs to run (see Figure 1-4). The file includes many different pieces of information, such as a display name that users see, a description of the app, the app's orientation (portrait, landscape, etc.), the file path to the app's icon, the app's capabilities (system features or devices that your app can use), and much more.

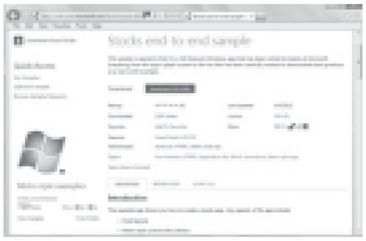
www.wiley.com/college/microsoft *or*
call the MOAC Toll-Free Number: 1+(888) 764-7001 (U.S. & Canada only)

Figure 1-5
A portion of the MSDN Developer Network Samples Web page



2. Scroll down and click the **HTML5** link in the left pane, currently near the bottom of the list.
3. Browse through the samples and find an app that displays "HTML5" after the app description, such as the stocks end-to-end sample. Click the link to the app.
4. The resulting page indicates which program is required to open and edit the app's files and which technologies are included in the app. The stocks end-to-end sample requires Visual Studio 12 and includes JavaScript and HTML5, as shown in Figure 1-6.

Figure 1-6
Viewing a sample app's Web page



5. Click the **Browse Code** link, click **StocksSample** in the left pane, click **html**, and then click **dashboard.html**. The HTML markup displays. Scroll through the markup to get a feel for the type of code you'll see many times in this book.

State management is the process of maintaining Web page information during multiple

Screen

2. Include the following nav tags and content within the <header> tag:

Images

Because the resources are stored on the cli ent's hard disk or device, the resources load faster when requested.

Using AppCache, a developer uses a text file called a "cache manifest" to specify the files a Web browser should cache offline. Even if a user presses the Refresh button offline, the app will load and work correctly. A cache manifest file looks similar to the following:

An example of markup for Previous-Home-Next links follows, with vertical bars after each
navigation item to separate it from the others visually:

```
<nav>
 <a href="http://www.example.com/Services">Previous</a> |
 <a href="http://www.example.com">Home</a> |
 <a href="http://www.example.com/About">Next</a>
</nav>
<br />
```

The links would appear in a Web page as shown in Figure 3-6.

Navigation is often displayed in a vertical list, which you'll learn how to do later in this
lesson.

Figure 3-6

Previous-Home-Next naviga tion with vertical bars separat ing each link

ADD THE NAV ELEMENT TO AN HTML DOCUMENT

GET READY. To add the nav element to an HTML document, perform the following
steps:

1. In your HTML editor or app development tool, open the **L3-MyPage.html** file
(if it's not already open) and save it as **L3-MyPage-nav.html** to create a new file.

```
<header>
 <h1>Selecting a Concert Style</h1>
 <nav>
 <a href="#symphonies">Symphonies</a> |
 <a href="#raves">Raves</a>
 </nav>
</header>
```

This navigation block will link to the Symphonies and Raves sections in the HTML document.

3. To make the links work, modify the Symphones and Raves <h1> heads as follows:

```
<h1><a id="symphonies">Symphonies</a></h1>
```

Certification Ready Alert

1.2

1.2

1.2

1.2

```
<h1><a id="raves">Raves</a></h1>
```

X

You'll learn more about using AppCache with JavaScript in Lesson 8.

requests for the same or different Web page. When a user first requests access to an application, the ***session state*** is created. The state ends when the user closes the session.

An alternative to the session state is the application state. The ***application state*** is created when the Web browser sends the first request for a Web page to the Web server, and it ends when the user closes the browser.

Persistent state information is data that an application needs after the session ends. Many Web applications need to store data (make it persistent) so that users can pick up where they left off when they return to the site.

Storing State Data Using Local and Session Storage

Hypertext Transport Protocol (HTTP) is the protocol that transfers data on the World Wide Web. It defines the actions Web servers and browsers take in response to commands by users. For example, when you enter a uniform resource locator (URL) in the address field in a browser, the browser sends an HTTP command to the Web server requesting the Web page. HTTP is a stateless protocol, which means it doesn't retain data from session to session. When you close a Web browser after visiting a Web site, the data is not saved.

To work around the limitations of HTTP protocol, developers historically have used ***cookies***, which are small files that contain information about the user and the Web site visited and are saved on the user's computer. When a user returns to a visited site, the browser sends the cookies back to the Web server. Cookies help a Web server "remember" a user and customize the user's experience on that site.

However, cookies have proven to be a security risk. In addition, if large amounts of data are involved, all of the data gets sent between the browser and server upon every request, which would cause a noticeable performance decrease to the user. In HTML5, developers can use the Web storage instead, which offers more flexibility, larger data sets, and better performance.

The localStorage method allows users to save larger amounts of data from session to session (persistent data), and there's no time limit as to how long the data exists. The sessionStorage method keeps data only for one session (until the browser is closed), which is also referred to as "per-tab storage."

Using these methods, specific data is transferred only when requested, so it's possible to store a relatively large amount of data without slowing down the connection or site.

APPCACHE FOR OFFLINE FILES

Another way to use Web storage is to store data locally when a user is offline. The Application Cache, or ***AppCache***, stores resources like images, HTML pages, CSS files,and JavaScript— data that would ordinarily be stored on a server.

```
index.html
stylesheet.css
images/dot.png
scripts/main.js
```

Step-by-Step Exercises
Cross Reference Reader Aid

www.wiley.com/college/microsoft *or*
call the MOAC Toll-Free Number: 1+(888) 764-7001 (U.S. & Canada only)

c02BuildingtheUserInterfacebyUsingHTML5TextGraphicsandMedia.indd Page 22 10/16/12 5:07 PM user-f494 /Users/F-440/Desktop

22 | **Lesson 2**

The busy Web site developers at Malted Milk Media have asked you to research new markup that's available in HTML5. They're particularly interested in graphics and multimedia-related tags. Your task is to learn all that you can about new HTML5 markup tags and prepare brief descriptions and provide examples of each.

■

THE BOTTOM LINE

Hypertext Markup Language (HTML) uses markup to describe content for display on a Web page. An element is the combination of tags and the content they enclose. You may need to use special characters on a Web page, which requires character encoding. Finally, every Web page requires the doctype declaration at the top of the page.

Reader Aid

Hypertext Markup Language (HTML) is called a markup language because you use it to describe (mark up) pieces of content to display on a Web page. A Web page with markup means it includes ***tags***, which are keywords that help to give an HTML page structure. (You'll learn more about tags shortly.) The key to using HTML is learning which tags to use and when. The combination of content, tags, and perhaps graphics, multimedia, and so on are what build a Web page.

You can easily identify an HTML document because it has an .htm or .html file extension. When a Web browser or mobile device such as a smartphone opens an HTML file, it ***renders*** (interprets and reproduces) the content of the page.

Basic Markup and Page Structure

Every HTML page includes tags. A tag is a keyword surrounded by angled brackets. Most tags come in pairs; one tag is called the opening or start tag, and the other is the closing or end tag. A tag pair is case sensitive—a closing tag must have the same case as the opening tag. A closing tag is identical to an opening tag except the closing tag includes a slash before the keyword.

Tags surround content and give it definition. For example, this markup creates a first-level heading:

```
<h1>Pet Care 101</h1>
```

HTML also uses some single tags, like <br /> for a line break and <hr /> for a horizontal line. In HTML 4, these tags are called ***empty tags*** because they don't require an end tag. HTML5 is less restrictive than HTML 4. You don't have to include end tags for all elements (although some elements still require start and end tags), and you can enter tags in uppercase or lowercase. However, this book uses start and end tags, and all lowercase for markup, for consistency.

There are many tags available for HTML pages. Some of the most commonly used tags are listed in Table 2-1. The first four—<html>, <head>, <title>, and <body>—are required

c03BuildingtheUserInterfacebyUsingHTML5OrganizationInputandValidation.indd Page 55 17/10/12 2:01 PM user-f396 F-403

Building the User Interface by Using HTML5: Organization, Input, and Validation | **55**

Take Note

on every Web page.

Figure 3-1

A comparison of document parts marked up in HTML 4.01 and HTML5

HTML 4.01

HTML 5

c07ManagingtheGraphicalInterfacebyUsingCSS.indd Page 163 9/17/12 10:01 AM F-400

Managing the Graphical Interface by Using CSS | **163**

```
<body>
 <div>A box example</div>
</body>
</html>
```

2. Save the file as **L7-box-exercise.html**. View the file in a Web browser, which should look similar to Figure 7-4.

Figure 7-4
A box with rounded corners

3. To add a drop shadow, add the following line to the style element:

```
box-shadow: 5px 5px 5px #999;
```

4. Save the file and view the results in a Web browser. The box should resemble Figure 7-5.

Figure 7-5
A box with rounded corners and a drop shadow

5. Leave the file, editing tool, and Web browser open if you complete the next exercise during this session.

APPLYING TRANSPARENCY

An opaque item does not let light pass through, whereas you can see through a transparent item. Even though the terms are opposite, by reducing the opacity of an item or increasing its

2.4

Table 3-1
New HTML5 markup tags for organizing content and creat ing structure

Notice how HTML5's semantic markup gives more specific meaning to parts of an HTML document, making the structure easier to understand.

Using Tags to Add Structure to an HTML Document

New HTML5 elements for structuring and organizing content in an HTML document include header, footer, section, nav, article, and aside.

Now that you understand semantic markup, let's look at several of the new HTML5 elements for organizing documents. Table 3-1 lists new HTML5 structure-related tags and describes them.

TAG D

<address> Defines an area for contact information for a page or section

<article> Defines an article, such as a magazine or newspaper article, blog post, or similar content

<aside> Defines content that's separate from but related to the page content; simi lar to a sidebar in book chapters and magazine articles

<details> Contains additional details pertinent to text around it; creates an interac tive widget a user can display or hide

<footer> Defines a footer for a document or section; may include the document author, contact information, copyright information, and links to terms
of use

<header> Defines a header for a document or section; may contain introductory con tent or navigation links

<hgroup> Groups headings and subheadings (using the <h1> to <h6> tags) for multi-level headings

<nav> Defines a block of navigation links

<section> Defines a section in a document, such as chapters, parts of a thesis, or parts of a Web page whose content is distinct from each other

<summary> Defines a visible heading for a details element; user can click to display or hide information

<wbr> Defines a possible line break; when a word is very long, or you're con cerned the browser will break a line at the wrong place, you can use the
<wbr> element to break the word or line appropriately

oto

hpkcotSi/GTVA

©

Figure 7-6

Transparency applied to an image

transparency, you eventually reach the same point.

Figure 7-6 shows the effect of ***transparency*** (or reduced ***opacity***) on an image. The original image is on the left; the image with a 50% transparency applied is on the right.

Original With transparency

ulll

3.4

The syntax for applying a transparency to an image or other element is: opacity: *value*

The value is a floating-point value between 0.0 (100% transparent) and 1.0 (100% opaque). To apply a 45% transparency, for example, you would use the value 0.55 (1.0 – 0.45).

:snoitartss

Easy-to-Read Tables

Photos

www.wiley.com/college/microsoft ***or***
call the MOAC Toll-Free Number: 1+(888) 764-7001 (U.S. & Canada only)

Although this example used the auto keyword, you can use any of the values for grid-rows and grid-colums as listed in Table 5-2.

The specification for grid template layouts is very much in draft format and isn't supported by any Web browsers at the time of this writing. However, you might come across grid templates on the MTA 98-375 exam. Therefore, you should check the latest W3C CSS Grid Template Layout Module specification when preparing to take the exam.

SKILLSUMMARY

- User interfaces can be clean and simple or more complex with several sections, buttons, and controls.
- Designing an interface that renders well on large PC screens and small mobile devices used to require a lot of markup and code. Today, the CSS Flexbox Box and Grid Layout models reduce the amount of code required for cross-device compatibility. Because the CSS specifications are not yet final, you'll need to use vendor prefixes before CSS property names to make everything work.
- Flexboxes are designed for toolbars, menus, forms, and similar elements in Web pages and applications. Grids are better suited to more complex designs.
- Both a flexbox and its contents can be configured to change size, horizontally and vertically, when the screen on which they're displayed changes size. You can also reverse the direction and order of flexboxes with one line of code.
- A flexbox can include child boxes that are flexible by height and width. You use the flex property to work with child boxes. The flex-flow property sets the flex-direction and flex-wrap properties of a flexbox (the parent box) at the same time.
- Grid layouts are similar to spreadsheets in that they use columns, rows, and cells, but you can create many different types of layouts that, in the end, don't look like a spreadsheet at all.
- You use the CSS properties display:grid (or display:inline-grid), grid-columns, and grid-rows to create grid structures. The size of columns and rows can be fixed or flexible.
- Flexboxes and grids are designed to scale proportionally.
- The flex-order property enables you to change the order of child items in a flexbox, rearranging them in any order you like without having to change them in the HTML markup.
- A grid template uses alphabetical characters to represent the position of items in a grid. You use the alpha characters with the grid-template, grid-rows, and grid-columns properties to create a grid into which data flows.

▪

Fill in the Blank

Complete the following sentences by writing the correct word or words in the blanks provided.

1. A is the portion of a Web site or application with which a user interacts.

2. In the original W3C CSS box model, the is the space between border and the content of the box.

Knowledge Assessment Questions

Understanding CSS Essentials: Layouts | 133

Although this example used the auto keyword, you can use any of the values for grid-rows and grid-colums as listed in Table 5-2.

The specification for grid template layouts is very much in draft format and isn't supported by any Web browsers at the time of this writing. However, you might come across grid templates on the MTA 98-375 exam. Therefore, you should check the latest W3C CSS Grid Template Layout Module specification when preparing to take the exam.

SKILLSUMMARY

- User interfaces can be clean and simple or more complex with several sections, buttons, and controls.
- Designing an interface that renders well on large PC screens and small mobile devices used to require a lot of markup and code. Today, the CSS Flexbox Box and Grid Layout models reduce the amount of code required for cross-device compatibility. Because the CSS specifications are not yet final, you'll need to use vendor prefixes before CSS property names to make everything work.
- Flexboxes are designed for toolbars, menus, forms, and similar elements in Web pages and applications. Grids are better suited to more complex designs.
- Both a flexbox and its contents can be configured to change size, horizontally and vertically, when the screen on which they're displayed changes size. You can also reverse the direction and order of flexboxes with one line of code.
- A flexbox can include child boxes that are flexible by height and width. You use the flex property to work with child boxes. The flex-flow property sets the flex-direction and flex-wrap properties of a flexbox (the parent box) at the same time.
- Grid layouts are similar to spreadsheets in that they use columns, rows, and cells, but you can create many different types of layouts that, in the end, don't look like a spreadsheet at all.
- You use the CSS properties display:grid (or display:inline-grid), grid-columns, and grid-rows to create grid structures. The size of columns and rows can be fixed or flexible.
- Flexboxes and grids are designed to scale proportionally.
- The flex-order property enables you to change the order of child items in a flexbox, rearranging them in any order you like without having to change them in the HTML markup.
- A grid template uses alphabetical characters to represent the position of items in a grid. You use the alpha characters with the grid-template, grid-rows, and grid-columns properties to create a grid into which data flows.

Fill in the Blank

Complete the following sentences by writing the correct word or words in the blanks provided.

1. A is the portion of a Web site or application with which a user interacts.

2. In the original W3C CSS box model, the is the space between border and the content of the box.

Skill Summary Matrix

c03BuildingtheUserInterfacebyUsingHTML5OrganizationInputandValidation.indd Page 86 17/10/12 2:01 PM user-f396 F-403

10. What is the format for the HTML5 tag that validates an email address?
a. <input label="email" name="URL">
b. <form id="email">
c. <label for="email">Email</label>
d. <input type = "email" name = "email">

True / False

Circle T if the statement is true or F if the statement is false.

T F 1. In a table, the tfoot element must appear before the tbody element.

T F 2. You can use numbers or letters for each item in an ordered list.
T F 3. You can specify the height of an input element using the size attribute.
T F 4. The label element displays the caption, or title, for a table.
T F 5. The nav element defi nes a block of navigation links.

▪

Scenario 3-1: Markup for a Newsletter Article

Sally Rowe, the document controller at Malted Milk Media, wants to publish a series of articles on the company intranet regarding document security and versioning. She needs to create a skeleton of the HTML5 markup for an article that will appear in the monthly online newsletter created by one of the Web developers. Each article will have a title and subtitle, several paragraphs of text, and her name and the article date in the footer. What should her article markup look like?

Scenario 3-2: Displaying Long Tables in HTML

Vince generates accounting reports for the VP of Finance at Momentum Strategies, a PR firm geared toward political campaigns. Vince regularly prints tables that are two- or three-pages long and delivers hard copies to senior management staff. He wants to publish them to a secure area of the company intranet, but the rows of data separate from the column headings and totals line at the end. He wants to know how to present the tables properly in HTML5. What do you tell him?

▪

Scenario 3-3: Creating a Glossary of Terms

Waylon is a student working on a term paper. His instructor requires each student to format the paper for display on the Web. Waylon wants to include a glossary of terms at the end of the paper but can't produce the right "look" using an unordered list. Which markup would be better suited for Waylon's glossary?

Scenario 3-4: Using Proper Input Types in a Web Form

Margie is creating and testing a Web form that includes an email field, a Web address field, and a zip code field, among others. When she has a few co-workers test the form, she finds they often enter the email address in the Web address field by mistake, and sometimes enter too many or too few numbers in the zip code field. She doesn't want to use a pattern expression because she says it's too complicated. What other input types can Margie use?

Proficiency Assessment

Competency Assessment

www.wiley.com/college/microsoft *or*
call the MOAC Toll-Free Number: 1+(888) 764-7001 (U.S. & Canada only)

This book uses particular fonts, symbols, and heading conventions to highlight important information or to call your attention to special steps. For more information about the features in each lesson, refer to the Illustrated Book Tour section.

CONVENTION M

THE BOTTOM LINE This feature provides a brief summary of the material to be covered in the section that follows.

CLOSE Words in all capital letters indicate instructions for opening, saving, or closing files or programs. They also point out items you should check or actions you should take.

CERTIFICATION READY This feature signals the point in the text where a specific certification objective is covered. It provides you with a chance to check your understanding of that particular MTA objective and, if necessary, review the section of the lesson where it is covered.

Reader aids appear in shaded boxes found in your text. Take Note provides helpful hints related to particular tasks or topics.

X These notes provide pointers to information discussed elsewhere in the textbook or describe interesting features of HTML5 that are not directly addressed in the current topic or exercise.

Alt + Alt A plus sign (+) between two key names means that you must press both keys at the same time. Keys that you are instructed to press in an exercise will appear in the font shown here.

Example Key terms appear in bold italic.

The *Microsoft Official Academic Course* programs are accompanied by a rich array of resources that incorporate the extensive textbook visuals to form a pedagogically cohesive package. These resources provide all the materials instructors need to deploy and deliver their courses. Resources available online for download include:

- **DreamSpark Premium** is designed to provide the easiest and most inexpensive developer tools, products, and technologies available to faculty and students in labs, classrooms, and on student PCs. A free 3-year membership is available to qualified MOAC adopters.

  Note: Microsoft Visual Studio and Micrsoft Expression can be downloaded from DreamSpark Premium for use by students in this course.

- The **Instructor Guides** contains Solutions to all the textbook exercises and Syllabi for vari ous term lengths. The Instructor Guides also includes chapter summaries and lecture notes. The Instructor's Guide is available from the Book Companion site (http://www.wiley.com/ college/microsoft).

- The **Test Bank** contains hundreds of questions in multiple-choice, true-false, short answer, and essay formats, and is available to download from the Instructor's Book Companion site (www.wiley.com/college/microsoft). A complete answer key is provided.

- A complete set of **PowerPoint presentations and images** are available on the Instructor's Book Companion site (http://www.wiley.com/college/microsoft) to enhance classroom presentations. Approximately 50 PowerPoint slides are provided for each lesson. Tailored to the text's topical coverage and Skills Matrix, these presentations are designed to con vey key concepts addressed in the text. All images from the text are on the Instructor's Book Companion site (http://www.wiley.com/college/microsoft). You can incorporate them into your PowerPoint presentations, or create your own overhead transparencies and handouts. By using these visuals in class discussions, you can help focus students' attention on key elements of technologies covered and help them understand how to use it effectively in the workplace.

- When it comes to improving the classroom experience, there is no better source of Wiley Faculty Network ideas and inspiration than your fellow colleagues. The **Wiley Faculty Network** con nects teachers with technology, facilitates the exchange of best practices, and helps to enhance instructional efficiency and effectiveness. Faculty Network activities include technology training and tutorials, virtual seminars, peer-to-peer exchanges of experi ences and ideas, personal consulting, and sharing of resources. For details visit www.WhereFacultyConnect.com.

DREAMSPARK PREMIUM—FREE 3-YEAR MEMBERSHIP AVAILABLE TO QUALIFIED ADOPTERS!

DreamSpark Premium is designed to provide the easiest and most inexpensive way for uni versities to make the latest Microsoft developer tools, products, and technologies available in labs, classrooms, and on student PCs. DreamSpark Premium is an annual membership program for departments teaching Science, Technology, Engineering, and Mathematics (STEM) courses. The membership provides a complete solution to keep academic labs, faculty, and students on the leading edge of technology.

Software available in the DreamSpark Premium program is provided at no charge to adopting departments through the Wiley and Microsoft publishing partnership.

And tools that professors can use to engage and inspire today's technology students.

Contact your Wiley rep for details.

For more information about the DreamSpark Premium program, go to:

https://www.dreamspark.com/

Note: Microsoft Visual Studio and Microsoft Expression can be downloaded from DreamSpark Premium for use by students in this course.

▪ Important Web Addresses and Phone Numbers

To locate the Wiley Higher Education Rep in your area, go to http://www.wiley.com/college and click on the "*Who's My Rep?*" link at the top of the page, or call the MOAC Toll Free Number: 1 + (888) 764-7001 (U.S. & Canada only).

To learn more about becoming certified and exam availability, visit www.microsoft.com/learning/mcp/mcp.

Additional Resources

Book Companion Web Site (www.wiley.com/college/microsoft)

The students' book companion site for the MOAC series includes any resources, exercise files, and Web links that will be used in conjunction with this course.

Wiley Desktop Editions

Wiley MOAC Desktop Editions are innovative, electronic versions of printed textbooks. Students buy the desktop version for up to 40% off the U.S. price of the printed text, and get the added value of permanence and portability. Wiley Desktop Editions provide students with numerous additional benefits that are not available with other e-text solutions.

Wiley Desktop Editions are NOT subscriptions; students download the Wiley Desktop Edition to their computer desktops. Students own the content they buy to keep for as long as they want. Once a Wiley Desktop Edition is downloaded to the computer desktop, students have instant access to all of the content without being online. Students can also print out the sections they prefer to read in hard copy. Students also have access to fully integrated resources within their Wiley Desktop Edition. From highlighting their e-text to taking and sharing notes, students can easily personalize their Wiley Desktop Edition as they are reading or following along in class.

About the Microsoft Technology Associate (MTA) Certification

Preparing Tomorrow's Technology Workforce

Technology plays a role in virtually every business around the world. Possessing the fundamental knowledge of how technology works and understanding its impact on today's academic and workplace environment is increasingly important—particularly for students interested in exploring professions involving technology. That's why Microsoft created the Microsoft Technology Associate (MTA) certification—a new entry-level credential that validates fundamental technology knowledge among students seeking to build a career in technology.

The Microsoft Technology Associate (MTA) certification is the ideal and preferred path to Microsoft's world-renowned technology certification programs, such as Microsoft Certified Solutions Developer (MCSD). MTA is positioned to become the premier credential for individuals seeking to explore and pursue a career in technology, or augment related pursuits such as business or any other field where technology is pervasive.

MTA Candidate Profile

The MTA certification program is designed specifically for secondary and post-secondary students interested in exploring academic and career options in a technology field. It offers students a certification in basic IT and development. As the new recommended entry point for Microsoft technology certifications, MTA is designed especially for students new to IT and software development. It is available exclusively in educational settings and easily integrates into the curricula of existing computer classes.

MTA Empowers Educators and Motivates Students

MTA provides a new standard for measuring and validating fundamental technology knowledge right in the classroom while keeping your budget and teaching resources intact. MTA helps institutions stand out as innovative providers of high-demand industry credentials and is easily deployed with a simple, convenient, and affordable suite of entry-level technology certification exams. MTA enables students to explore career paths in technology without requiring a big investment of time and resources, while providing a career foundation and the confidence to succeed in advanced studies and future vocational endeavors.

In addition to giving students an entry-level Microsoft certification, MTA is designed to be a stepping stone to other, more advanced Microsoft technology certifications, like the Microsoft Certified Solutions Developer (MCSD) certification.

Delivering MTA Exams: The MTA Campus License

Implementing a new certification program in your classroom has never been so easy with the MTA Campus License. Through the purchase of an annual MTA Campus License, there's no more need for ad hoc budget requests and recurrent purchases of exam vouchers. Now you can budget for one low cost for the entire year, and then administer MTA exams to your students and other faculty across your entire campus where and when you want.

The MTA Campus License provides a convenient and affordable suite of entry-level technology certifications designed to empower educators and motivate students as they build a foundation for their careers.

The MTA Campus License is administered by Certiport, Microsoft's exclusive MTA exam provider.

To learn more about becoming a Microsoft Technology Associate and exam availability, visit www.microsoft.com/learning/mta.

www.wiley.com/college/microsoft *or*
call the MOAC Toll-Free Number: 1+(888) 764-7001 (U.S. & Canada only)

Activate Your FREE MTA Practice Test!

Your purchase of this book entitles you to a free MTA practice test from GMetrix (a $30 value). Please go to www.gmetrix.com/mtatests and use the following validation code to redeem your free test: **MTA98-375-3324376A6A85.**

The **GMetrix Skills Management System** provides everything you need to practice for the Microsoft Technology Associate (MTA) Certification.

Overview of Test features:

• Practice tests map to the Microsoft Technology Associate (MTA) exam objectives

• GMetrix MTA practice tests simulate the actual MTA testing environment • 50+ questions per test covering all objectives

• Progress at own pace, save test to resume later, return to skipped questions

• Detailed, printable score report highlighting areas requiring further review

To get the most from your MTA preparation, take advantage of your free GMetrix MTA Practice Test today!

For technical support issues on installation or code activation, please email support@gmetrix.com.

MOAC Toll-Free Number: 1+(888) 764-7001 (U.S. & Canada only)

MOAC MTA Technology Fundamentals Reviewers

We'd like to thank the many reviewers who pored over the manuscript and provided invaluable feedback in the service of quality instructional materials:

Yuke Wang, University of Texas at
Dallas Palaniappan Vairavan, Bellevue
College
Harold "Buz" Lamson, ITT Technical
Institute Colin Archibald, Valencia Community
College Catherine Bradfield, DeVry University
Online Robert Nelson, Blinn College

Kalpana Viswanathan, Bellevue College
Bob Becker, Vatterott College
Carol Torkko, Bellevue College
Bharat Kandel, Missouri Tech
Linda Cohen, Forsyth Technical Community College
Candice Lambert, Metro Technology Centers Susan
Mahon, Collin College
Mark Aruda, Hillsborough Community College

Claude Russo, Brevard Community College
Heith Hennel, Valencia College
Adrian Genesir, Western Governors
University Zeshan Sattar, Zenos
Douglas Tabbutt, Blackhawk Technical
College David Koppy, Baker College
Sharon Moran, Hillsborough Community College
Keith Hoell, Briarcliffe College and Queens College—CUNY
Mark Hufnagel, Lee County School District
Rachelle Hall, Glendale Community College

Scott Elliott, Christie Digital Systems, Inc.
Gralan Gilliam, Kaplan
Steve Strom, Butler Community College
John Crowley, Bucks County Community College
Margaret Leary, Northern Virginia Community College
Sue Miner, Lehigh Carbon Community College Gary
Rollinson, Cabrillo College
Al Kelly, University of Advancing
Technology Katherine James, Seneca College
David Kidd, Western Governors University

Lesson 1: Managing the Application Life Cycle 1

Lesson 2: Building the User Interface by Using HTML5: Text, Graphics, and Media 21

Lesson 3: Building the User Interface by Using HTML5: Organization, Input, and Validation 53

Lesson 4: Understanding CSS Essentials: Content Flow, Positioning, and Styling 87

Lesson 5: Understanding CSS Essentials: Layouts

110 **Lesson 6:** Managing Text Flow by Using CSS

137

Lesson 7: Managing the Graphical Interface by Using CSS

159 **Lesson 8:** Understanding JavaScript and Coding

Essentials 189

Lesson 9: Creating Animations, Working with Graphics, and Accessing Data 215

Lesson 10: JavaScript Coding for the Touch Interface, Device and Operating System Resources, and More 243

Appendix 268

Index 269

Lesson 1: Managing the Application Life Cycle 1

Exam Objective Matrix 1
Key Terms 1
Understanding Platform

Fundamentals 2 What's New in HTML5? 3
Creating Apps 5
Exploring Packaging and the Runtime Environment 7 Understanding the Host Process 7
Understanding the App Package and App Container 8 Understanding Credentials and Permission Sets 10 **Understanding and Managing Application States** 10
Storing State Data Using Local and Session Storage 11 AppCache for Offline Files 11
Understanding Touch Interfaces and Gestures 12 Leveraging Existing HTML5 Skills

and Content for Slate/Tablet Applications 13

Debugging and Testing HTML5 Apps 13 Validating HTML5 Code 14
Validating a Package 14

Publishing an Application to a Store 16 **Skill Summary** 17

Knowledge Assessment 18

Competency Assessment 20

Proficiency Assessment 20

Lesson 2: Building the User Interface by Using HTML5: Text, Graphics, and Media 21

Exam Objective Matrix 21

Key Terms 21

Understanding the Essentials of HTML 22 Basic Markup and Page Structure 22
Using Attributes 23
Nesting Elements 24
Understanding Entities 24
Understanding the Doctype 25
Exploring the Markup of a Simple Web Page 26

Choosing and Configuring HTML5 Tags to Display Text Content 29
Text Elements from HTML 4 with New Meaning or Functionality 29
New Text Elements in HTML5 31
Text Elements Not Used in HTML5 32

Choosing and Configuring HTML5 Tags to Display Graphics 34
Using the figure and figcaption Elements 35
Creating Graphics with Canvas 38
Canvas Basics 39
Creating an Outline of a Shape 40
Providing an Alternate Image or Text for Older Browsers 41 Creating Graphics with SVG 42
When to Use Canvas Instead of SVG 44

Choosing and Configuring HTML5 Tags to Play Media 45
Understanding and Using Video Tags 45
Understanding and Using Audio Tags 47

Skill Summary 49

Knowledge Assessment 49

Competency Assessment 51

Proficiency Assessment 52

Lesson 3: Building the User Interface by Using HTML5: Organization, Input, and Validation 53

Exam Objective Matrix 53

Key Terms 53

Choosing and Configuring HTML5 Tags to Organize Content and Forms 54
Understanding Semantic HTML 54
Using Tags to Add Structure to an HTML Document 55 The header and footer Elements 56
The section Element 57
The nav Element 59
The article Element 61
The aside Element 61
Using Tags to Create Tables and Lists 64
Creating Tables 64
Creating Lists 69

Choosing and Configuring HTML5 Tags for Input and Validation 72
Understanding Input and Forms 73
Exploring Form Creation, Input Attributes, and Values 77 Understanding Validation 81

Skill Summary 83

Knowledge Assessment 84

Competency Assessment 86

Proficiency Assessment 86

Lesson 4: Understanding CSS Essentials:

Content Flow, Positioning, and Styling 87

Objective Domain Matrix 87
Key Terms 87
Understanding CSS Essentials 87
Using the Appropriate Tools 88
Exploring the Link between HTML and CSS 89
Separating Content from Style 91
Understanding Selectors and Declarations 92
Understanding Fonts and Font Families 94
Managing Content Flow 96
Positioning Individual Elements 99
Applying Float Positioning 99
Applying Absolute Positioning 100
Managing Content Overflow 102
Understanding Scrolling Overflow 102
Understanding Visible Overflow and Hidden Overflow 104 **Skill Summary** 105
Knowledge Assessment 106
Competency Assessment 108
Proficiency Assessment 109

Lesson 5: Understanding CSS Essentials: Layouts 110

Exam Objective Matrix 110
Key Terms 110
Arranging User Interface (UI) Content by Using CSS 111
Using Flexbox for Simple Layouts and Using Grid for Complex Layouts 112
Using a Flexible Box to Establish Content Alignment, Direction, and Orientation 114
Work with Flexboxes and Flexbox Items 116
Applying Proportional Scaling within a Flexbox 116 Changing the Direction of Child Items in a Flexbox 122 Ordering and Arranging Content 126
Using Grid Layouts to Establish Content Alignment, Direction, and Orientation 128 Creating a Grid Using CSS Properties for
Rows and Columns 130
Understanding Grid Templates 132
Skill Summary 133
Knowledge Assessment 133
Competency Assessment 135
Proficiency Assessment 136

Lesson 6: Managing Text Flow by Using CSS 137

Exam Objective Matrix 137
Key Terms 137
Managing the Flow of Text Content by Using CSS 137 **Understanding and Using Regions to Flow Text Content between Multiple Sections** 139 Flowing Content through Containers Dynamically 140 Overflowing Text 142
Microsoft's Implementation of CSS Regions 142
Using Columns and Hyphenation to Optimize the Readability of Text 145
Creating Columns 146
Using Hyphenation 150
Using CSS Exclusions to Create Text Flow around a Floating Object 152
Skill Summary 155
Knowledge Assessment 155
Competency Assessment 157
Proficiency Assessment 158

Lesson 7: Managing the Graphical Interface by Using CSS 159

Exam Objective Matrix 159
Key Terms 159
Managing the Graphical Interface with CSS 159 Creating Graphics Effects 160
Creating Rounded Corners 160
Creating Shadows 161
Applying Transparency 163
Applying Background Gradients 164

Understanding Typography and the Web Open Font Format 166
Applying 2D and 3D Transformations 167

2D Translation 168
2D Scaling 169
2D and 3D Rotation 171
2D and 3D Skewing 172
Understanding 3D Perspective, Transitions, and
Animations 173 Applying SVG Filter Effects 179
Using Canvas to Enhance the GUI 182

Skill Summary 185

Knowledge Assessment 186

Competency Assessment 188

Proficiency Assessment 188

Lesson 8: Understanding JavaScript and Coding Essentials 189

Objective Domain Matrix 189

Key Terms 189

Managing and Maintaining JavaScript
189 Creating and Using Functions 193
Using jQuery and Other Third-Party
Libraries 197 **Updating the UI by Using**
JavaScript 199 Locating and Accessing
Elements 201
Listening and Responding to Events 203
Showing and Hiding Elements 206
Updating the Content of Elements 208
Adding Elements 209

Skill Summary 211

Knowledge Assessment 212

Competency Assessment 214

Proficiency Assessment 214

Lesson 9: Creating Animations, Working with Graphics, and Accessing Data 215

Objective Domain Matrix 215

Key Terms 215

Coding Animations by Using
JavaScript 216 Creating Animations 216

Working with Images, Shapes,
and Other Graphics 219
Manipulating the Canvas with JavaScript 220

Sending and Receiving Data 224
Transmitting Complex Objects and
Parsing 227 **Loading and Saving Files**
229
Using the Application Cache (AppCache) 231
Understanding and Using Data Types 233

Using JavaScript to Validate User Form
Input 233 **Understanding and Using Cookies**
235

Understanding and Using Local
Storage 237 **Skill Summary** 239

Knowledge Assessment 240

Competency Assessment 242

Proficiency Assessment 242

Lesson 10: JavaScript Coding for the Touch Interface, Device and Operating System Resources, and More 243

Exam Objective Matrix 243

Key Terms 243

Responding to the Touch Interface
244 Capturing and Responding to
Gestures 246

Coding Additional HTML5 APIs 249
Coding to Capture GeoLocation 249
Understanding Web Workers 252
Understanding WebSockets 255
Using File API for File Uploads 258

Accessing Device and Operating
System Resources 260
Accessing In-Memory Resources 260
Accessing Hardware Capabilities 262
Understanding Global Positioning System (GPS) 263
Understanding Accelerometer 263
Accessing a Camera 263

Skill Summary 264

Knowledge Assessment 265

Competency Assessment 267

Proficiency Assessment 267

Appendix 268

Index 269

www.wiley.com/college/microsoft *or*
call the MOAC Toll-Free Number: 1+(888) 764-7001 (U.S. & Canada only)

LESSON

Managing the Application Life Cycle

EXAM OBJECTIVE MATRIX

S /Concepts	MTA Exam Objective	MTA Exam Objective N
Understanding Platform Fundamentals	Understand the platform fundamentals.	1.1
Understanding and Managing Application States	Manage the state of an application.	1.2
Understanding Touch Interfaces and Gestures	Understand the platform fundamentals.	1.1
	Debug and test an HTML5-based touch-enabled application.	1.3
Debugging and Testing HTML5 Apps	Debug and test an HTML5-based touch-enabled application.	1.3
Publishing an Application to a Store	Publish an application to a store.	1.4

app container
app package
AppCache
application programming interface (API)
application state
Cascading Style Sheets (CSS)
cookies
debugging
gesture
Hypertext Markup Language (HTML)
Hypertext Transport Protocol (HTTP)
HTML5
identity permissions
JavaScript
launcher icon
localStorage
markup language
media queries

Metro-style user interface (UI)
namespace
permission sets
persistent state information
platform-independent
scripting language
session state

sessionStorage
touch event
touch-screen simulator or emulator
validator
Windows Runtime (WinRT)
Windows Store
World Wide Web Consortium (W3C)

You are the new intern at Malted Milk Media, a creative agency that creates rich media applications for clients. The company will soon begin using HTML5 in its projects. Your manager asked you to research HTML5 and its related technologies and present a report on significant changes from HTML 4.01 to HTML5. You must also include information on how HTML5 can be used for creating touch-screen applications, such as those for PCs, slates, tablets, and smartphones.

■

THE BOTTOM LINE

1.1

HTML5 is the latest HTML standard and a family of technologies that includes HTML, CSS, and JavaScript. Although the HTML5 standard won't be finalized for a few years, most modern Web browsers already support HTML5 elements, and HTML5 app development for Web and mobile device browsers is well underway.

Hypertext Markup Language (HTML) is the language you use to describe Web pages. It is a ***markup language***, not a programming language, which means HTML uses markup tags such as `<body>` and `<h1>` to describe parts of a Web page. An HTML file doesn't "run" like a program. Instead, an HTML file is interpreted by a browser to display a Web page based on the tags.

Since 1999, HTML 4.01 has been the standard for Web pages, but the world has changed quite a bit since then. Web users want richer Web applications that incorporate audio, video, and a lot of interactivity on the Web sites they visit. And with the surge in popularity of mobile devices like slates, tablets, and smartphones, users want to experience the same richness and interactivity in mobile applications regardless of which device they choose.

All of this has prompted the need for a new standard, which will be ***HTML5***. The ***World Wide Web Consortium (W3C)*** is the main standards body developing specifications for HTML5, which should be finalized in 2014. The HTML5 logo is shown in Figure 1-1.

*

W3C® is a trademark (registered in numerous countries) of the World Wide Web Consortium; marks of W3C are registered and held by its host institutions MIT, ERCIM, and Keio.

Figure 1-1
The HTML5 logo

*

Even though HTML5 is still under development, most of the major Web browsers such as Microsoft Internet Explorer, Google Chrome, Mozilla Firefox, Opera, and Apple Safari support many HTML5 elements.

This work is attributed to the W3C.

An important point to remember about HTML5 is that it is both a standard and a combination or family of new HTML markup tags, CSS, JavaScript, and other related technologies. ***Cascading Style Sheets (CSS)*** defines styles for HTML in a separate file, so you can easily change fonts, font sizes, and other attributes in a CSS file and the changes are reflected across all HTML files that ref erence the CSS file. The latest version of CSS is CSS3. ***JavaScript*** is a ***scripting language*** (a pro gramming language that uses scripts and requires no compiler) that adds interactivity to Web pages.

Although you can use HTML5, CSS3, and JavaScript to create Web pages, you can also use the combination to develop client applications (apps) that run on touch-enabled devices like

*

You'll learn about CSS3 and JavaScript in depth in later lessons.

*

PCs, slates, tablets, and smartphones. Essentially, the same technologies developers use to build Web pages are now beginning to be used to build applications that run on different devices.

HTML5 is also ***platform-independent***. That means you can create apps using the HTML5 family of technologies that can run on different desktop and mobile device operating systems, such as Microsoft Windows, Internet Explorer, and Windows Phone. You can also run them in Mac OS X, Android, iOS, and Blackberry OS. Because HTML5 is built on an open standard, users of HTML5 apps do not have to download a plug-in or use devices that have plug-in support. Instead, you can use any Web browser, whether on your PC or mobile device, and get the same rich Web experience.

Finally, an important part of app development in the Windows environment is the ***Metro style user interface (UI)***, which is the UI used by the latest Microsoft Windows version: Windows 8. The Metro style UI includes features like a clean, uncluttered look and feel, use of the full screen, large hubs (graphical buttons), and a focus on lateral scrolling, to name a few. See Figure 1-2 as an example.

Figure 1-2

The Windows 8 Start screen is a Metro style UI

Because this book helps prepare you for the Microsoft Technology Associate (MTA) 98-375 certification exam, HTML5 Application Development Fundamentals, examples in the lessons use Microsoft tools as much as possible. However, the exam does not focus on a specific set of tools. You can work with the HTML5 family using many different tools from many different companies. Even a simple text editor like Notepad or Notepad++ does the trick when working with HTML markup, CSS, and JavaScript. You need the more comprehensive tools, like Visual Studio, when debugging a lot of code, packaging apps for distribution, and similar tasks. Free development tools for creating Metro style apps are available at http://bit.ly/K8nkk1.

What's New in HTML5?

The HTML5 family includes many new markup tags and technologies like media queries, geolocation, Modernizr, and much more. These technologies add a lot of functionality to HTML-based apps and help make the finished product more stylish.

The following is a short list of these new features and brief descriptions:

- **Audio and video tags:** Embeds audio and video multimedia using the HTML5 markup tags `<audio>` and `<video>`.
- **Canvas:** An HTML5 element that creates a container for graphics, and uses JavaScript to draw the graphics as needed.
- ***Media queries:*** A CSS3 feature that detects the user's type of screen and sizes the output accordingly.
- **New application programming interfaces (APIs):** Give apps access to a plethora of resources, such as files, webcams, and hardware-accelerated animations.
- **Geolocation:** Uses JavaScript to detect the location (geographic positioning) of a client device, whether it's a Windows Phone, Android phone, or a PC.
- **Modernizr:** A JavaScript library that helps you deliver the new capabilities of HTML5 and CSS3 in older browsers.

This is a small sampling of the available features and technologies. You'll learn how to use many of these in lessons throughout the course.

EXPLORE THE HTML5 STANDARD

GET READY. To learn about the HTML5 standard, perform the following steps:

1. Go to the W3C Web site at http://www.w3.org/TR/html5/.
2. Read the content on the first few pages, until you reach the Table of Contents and then address the following questions:
   - What is the latest published version of the standard?
   - Which working group is responsible for the specification?
   - What is the name of the Web page that tracks bugs, and what are three bugs that have not yet been addressed?
   - What is the name of the Web page that tracks outstanding issues, and what are three issues that have not yet been addressed?
3. On the main HTML5 Web page, spend about 15 minutes browsing the remainder of the

page to become familiar with the topics.

EXPLORE APP DEVELOPER RESOURCES

GET READY. To learn about app develop resources provided by Microsoft, perform the following steps:

1. Go to the Beginner Developer Learning Center Web site at http://msdn.microsoft.com/en-us/beginner/default.aspx.
2. Click the **Getting started with Windows Metro style apps development** link. On the resulting page, browse the information. Which technologies or development tools can you use to create Metro style apps?
3. Go to the HTML/CSS for Metro style apps Web page at http://bit.ly/N48F0L.
4. Click the **HTML and DOM reference** link and then answer the following question:
   - What is the Document Object Model (DOM) and what is its significance to Metro Style apps?
5. Return to the **HTML/CSS for Metro style apps** page at http://bit.ly/N48F0L. Click the **Cascading style sheets reference** link and then address the following:
   - Name three elements of Web pages that are controlled using CSS.
6. Go to the MSDN site at http://bit.ly/Hd9uzt. Browse the information to become familiar with the site.

MORE INFORMATION

The Web addresses used in the previous exercises are repeated here for your convenience—consider bookmarking them for later use. You can find information on the HTML5 standard on the W3C Web site at http://www.w3.org/TR/html5/. One Web site that provides HTML5 tutorials is at http://www.w3schools.com/html5/default.asp. Microsoft provides useful information to new developers at the Beginner Developer Learning Center (http://msdn.microsoft.com/en-us/beginner/default.aspx) and the MSDN site at http://bit.ly/Hd9uzt.

Creating Apps

HTML5 apps are much easier to develop than similar apps that use other technologies such as Adobe Flash and Microsoft Silverlight. HTML5, CSS, and JavaScript are interpreted languages, which means they do not require compilation. You can debug the code within a browser, make quick edits, and then refresh the browser window to see the result of the changes.

Although you can create a lot of code using a simple text or HTML editor, if you want to package and deploy your app, you'll need to use an application development tool such as Microsoft Visual Studio.

Figure 1-3

General steps for creating an app

Plan

Design UI Build App

Update Manifest Write

Test and Debug Package

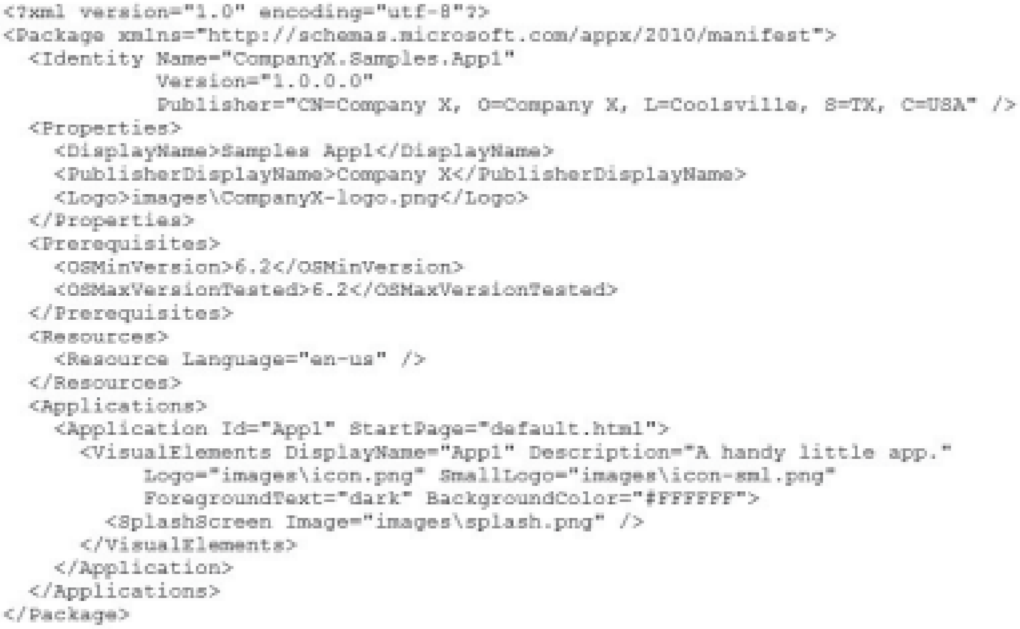
Validate

Code

Deploy

In this section, you will learn the general steps involved in creating an app. Figure 1-3 illustrates the steps, which are explained as follows:

- **Plan your project:** Think about the type of app you want to create. Whatever you choose, it's best to keep it simple while you're first learning HTML5 app development. After you decide on the main action of your app, create an outline of the general flow of the application from start to finish. Also determine the type of user interactivity you want to include, such as a touch interface, whether you need to save data outside of the app, and whether the app should connect to other apps or services (such as an RSS feed).
- **Design a UI:** When designing the user interface, determine how you want the app to appear to users. Group the content in a way that makes logical sense. Begin listing the commands you will need for the app to run as expected, and gather images and multimedia clips if necessary. At a minimum, today's apps require a ***launcher icon***, which represents your app.
- **Update the app manifest:** Every app requires a manifest file. The manifest file describes properties of the app and what the app needs to run (see Figure 1-4). The file includes many different pieces of information, such as a display name that users see, a description of the app, the app's orientation (portrait, landscape, etc.), the file path to the app's icon, the app's capabilities (system features or devices that your app can use), and much more.



```
<?xml version="1.0" encoding="utf-8"?>
<Package xmlns="http://schemas.microsoft.com/appx/2010/manifest">
  <Identity Name="CompanyX.Samples.App1"
            Version="1.0.0.0"
            Publisher="CN=Company X, O=Company X, L=Coolsville, S=TX, C=USA" />
  <Properties>
    <DisplayName>Samples App1</DisplayName>
    <PublisherDisplayName>Company X</PublisherDisplayName>
    <Logo>images\CompanyX-logo.png</Logo>
  </Properties>
  <Prerequisites>
    <OSMinVersion>6.2</OSMinVersion>
    <OSMaxVersionTested>6.2</OSMaxVersionTested>
  </Prerequisites>
  <Resources>
    <Resource Language="en-us" />
  </Resources>
  <Applications>
    <Application Id="App1" StartPage="default.html">
      <VisualElements DisplayName="App1" Description="A handy little app."
           Logo="images\icon.png" SmallLogo="images\icon-sml.png"
           ForegroundText="dark" BackgroundColor="#FFFFFF">
        <SplashScreen Image="images\splash.png" />
      </VisualElements>
    </Application>
  </Applications>
</Package>
```

Figure 1-4

An example of a manifest file

- **Write code:** During this phase, you compose the code for your application, which might include a combination of HTML, CSS, and JavaScript.
- **Build the app:** Using an app development tool such as Visual Studio, convert your code and other resources into an actual application.
- **Debug and test:** You must test your app thoroughly and fix any problems that appear. If the app uses a touch interface, it's highly important to test the app on a touch device or use a touch emulator.

- **Package:** Packaging an app creates a container that holds all of the various files required by the app, such as JavaScript, images, and so on.
- **Validate:** Validating your app means running it through a validation program to ensure nothing is missing,
- **Deploy:** Upload your app to a marketplace such as the Windows Store.

Apps that you plan to deploy to many people, especially through a marketplace such as an app store, must be reliable and secure. Many apps are also designed to run on multiple operating systems. Be sure you have tested your app thoroughly and validated it with the proper tools. You should also consider providing technical support for more complex apps.

PREPARE FOR APPLICATION DEVELOPMENT

GET READY. To prepare to work with HTML5 and develop apps, perform the following steps: **1.** Look for sources of free, non-copyrighted images on the Web. Even if you're capable

of creating many of your own graphics, having resources to draw from will come in handy.

2. To work with HTML5, CSS, and JavaScript files, download and install a text or HTML editor, such as Notepad++. (Just search for Notepad++ using a Web browser.) A more full-featured tool is Visual Studio Express for Web, available from the Visual Studio Web site at http://bit.ly/eBUygk. Express for Web lets you open your files into a Web browser with one click and provides lots of templates to help you create files quickly.

*

3. Ensure you have the latest versions of your browser installed.

4. To create apps for the Windows Store, download Microsoft Visual Studio 2012 Express for Windows 8 from http://bit.ly/K8nkk1 and install it. The program requires Windows 8 to be installed.

Exploring Packaging and the Runtime Environment

The Windows Runtime environment is the foundation of the Windows 8 operating system and provides functionality to Metro-style apps.

When an application is launched, it's considered to be in a runtime environment (RTE). This is the environment in which developers test their applications, and where users run the apps. Windows has its own runtime environment, called ***Windows Runtime (WinRT)***.

The WinRT is the foundation of the Windows 8 operating system, and is made up of layers that provide functionality to Metro-style apps and the Windows shell. WinRT supports apps written in different languages that use the Metro UI.

The Windows Core layer is at the base. This layer includes the Windows kernel, services, and user mode. Moving up, the Windows Runtime Core includes additional services like memory management and globalization. Above the Windows Runtime Core are layers related to devices, along with media, networking, local and remote storage, and more. The UI layer supports HTML5 apps, along with others.

The WinRT works with C#, C++, Visual Basic, and JavaScript. You can build Metro style apps with the WinRT and Windows Library for JavaScript APIs. An ***application programming interface (API)*** is simply a list of instructions letting a program communicate with another program. In a Web app, an API enables a Web browser or a Web server to communicate with other programs. There are hundreds of APIs available for many different uses.

The Document Object Model (DOM) is an important API to keep in mind. The DOM is designed for HTML and Extensible Markup Language (XML), and allows programs and scripts to update content, structure, and styles on the fly—essentially anything in an HTML or XML file can be modified. The DOM is neither HTML nor JavaScript, but it ties them together.

The Windows Library for JavaScript includes JavaScript and CSS files which developers can use to create Metro style apps more easily and quickly. You use the library along with HTML, CSS, and the WinRT to create apps.

The runtime environment is responsible for access to devices, media, networking, local and remote storage, and other items. A developer can use APIs and the runtime environment to request access to user devices within an app. In a Windows 8 app, for example, the device could be a keyboard, mouse, touchpad, printer, webcam, or microphone.

MORE INFORMATION

For more information about Windows Runtime, visit the “HTML, CSS, and JavaScript features and differences” Web page at http://bit.ly/xrofoB.

UNDERSTANDING THE HOST PROCESS

Whether an app is a Web app or whether it’s created for Windows, an app requires a runtime host to start it. For example, when you start Internet Explorer, a host process in the operating system controls the overall execution of the browser. (A “process” is simply a program that’s being executed.) In this case, each browser tab gets its own process, so if you have three tabs open, the system has three processes running for each of those tabs.

When you run a Metro style app that was created with JavaScript, Internet Explorer renders the HTML much like when you browse to a Web page, but the browser is hosted by a different process, called WWAHost.exe. This process runs the app inside of an app container. (You’ll learn about app containers in the next section.) WWAHost passes the HTML, CSS, and JavaScript into the default.html page that is the start of your app.

1.1

UNDERSTANDING THE APP PACKAGE AND APP CONTAINER

The purpose of an app package is for ease of distribution and deployment. Application pack aging bundles an app’s files and folders into a distributable package. An app container ensures the application runs in its own memory space and doesn’t corrupt the operating system.

Application packaging is the process of bundling an application and its various files into a distributable file, making it easy to deploy the app. The ***app package*** is the result of the pack aging process. Packaging is similar to archiving a folder that contains files and subfolders. It would be difficult to send all of the files and subfolders to someone as is, but the job is much easier when you compress everything into a single archive file. App development packages like Visual Studio provide the functionality to create app packages. A user acquires an app package, usually from an online app store, and installs it on a PC or device. The application executes in a runtime ***app container***, which means a separate memory space. An app con tainer prevents corruption of the operating system if the application fails for some reason and enables a user to cleanly uninstall the app.

Some things you should know about packages are as follows:

- A package may contain Web pages, code, database tables, and procedures. When a package has a user interface, it’s referred to as an application.
- A package can contain other packages.
- You can move one or more elements in or out of a package. Because a package is in its own container, if you move a package, then everything in the package moves as a unit.
- A user can install, upgrade, or remove a package.

A single package can have a lot of functionality. To keep all of the components separated so they don’t conflict, a package defines a ***namespace***. Think of a namespace as a work area for related objects (pages, code, etc.).

1.1

To keep things orderly, the code running in a Metro style app container is restricted to certain actions, by default. If you want your app to access a device, another app, the Internet, or any thing outside of itself, you must declare (specify) the interaction in the app manifest. These declarations are located in the Capabilities section of the manifest. When the end user installs the app, the user must give permission for the requested access.

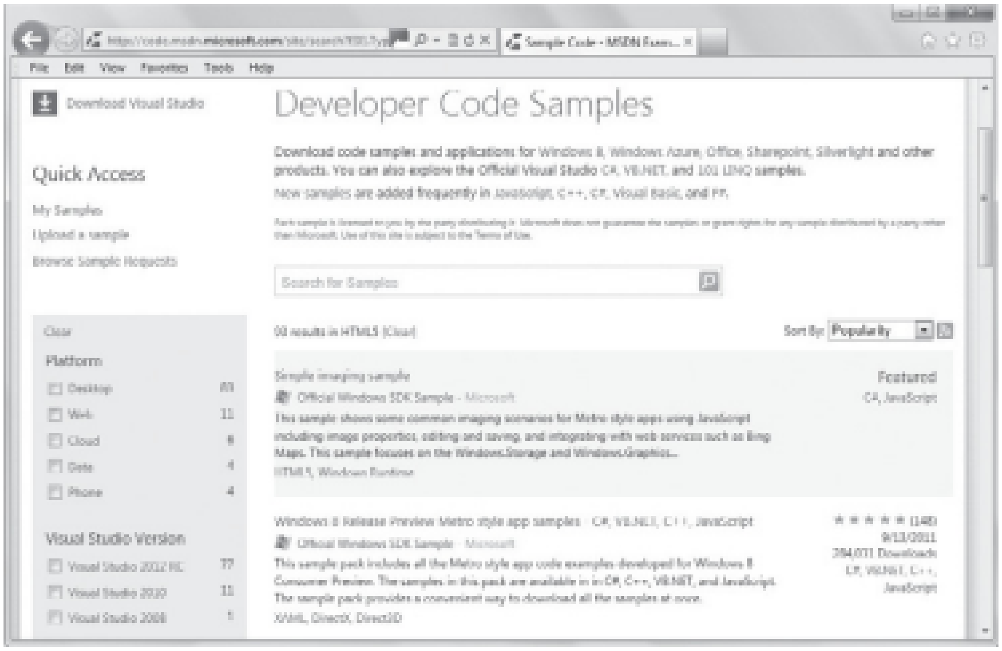
Metro style apps use contracts, which are essentially agreements, and something called exten sions when creating interactions between apps. WinRT APIs handle the communication between the apps.

EXPLORE APP SAMPLES

GET READY. To explore the kinds of sample apps that are available for download, perform the following steps:

1. Go to the MSDN Developer Network Samples Web

page (see Figure 1-5) at http://bit.ly/H57ZVh. Microsoft provides a wide range of sample apps and code samples, which you can download and open in an app development tool like Visual Studio. You can also view the code for many sample apps online.



Developer Code Samples

Download code samples and applications for Windows 8, Windows Azure, Office, Sharepoint, Silverlight and other products. You can also explore the Official Visual Studio C#, VB.NET, and 101 LINQ samples.

New samples are added frequently in JavaScript, C++, C#, Visual Basic, and F#.

Each sample is licensed to you by the party distributing it. Microsoft does not guarantee the samples or grant rights for any sample distributed by a party other than Microsoft. Use of this site is subject to the Terms of Use.

93 results in HTML5 (Clear)

Simple imaging sample
Official Windows SDK Sample - Microsoft
This sample shows some common imaging scenarios for Metro style apps using JavaScript including image properties, editing and saving, and integrating with web services such as Bing Maps. This sample focuses on the Windows.Storage and Windows.Graphics...
HTML5, Windows Runtime
Featured
C#, JavaScript

Windows 8 Release Preview Metro style app samples C#, VB.NET, C++, JavaScript
Official Windows SDK Sample - Microsoft
This sample pack includes all the Metro style app code examples developed for Windows 8 Consumer Preview. The samples in this pack are available in in C#, C++, VB.NET, and JavaScript. The sample pack provides a convenient way to download all the samples at once.
XAML, DirectX, Direct2D
★★★★★ (148)
9/13/2011
284,071 Downloads
C#, VB.NET, C++, JavaScript

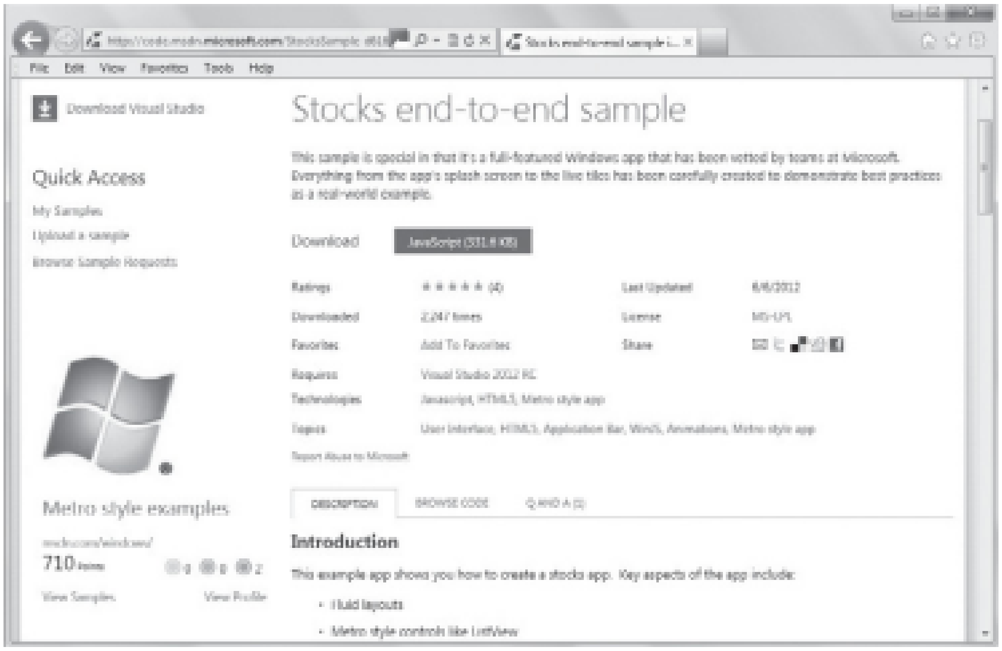
Figure 1-5

A portion of the MSDN Developer Network Samples Web page

2. Scroll down and click the **HTML5** link in the left pane, currently near the bottom of the list.

3. Browse through the samples and find an app that displays “HTML5” after the app description, such as the stocks end-to-end sample. Click the link to the app.

4. The resulting page indicates which program is required to open and edit the app’s files and which technologies are included in the app. The stocks end-to-end sample requires Visual Studio 12 and includes JavaScript and HTML5, as shown in Figure 1-6.



Stocks end-to-end sample

This sample is special in that it's a full-featured Windows app that has been vetted by teams at Microsoft. Everything from the app's splash screen to the live tiles has been carefully created to demonstrate best practices as a real-world example.

Download JavaScript (331.8 KB)

Ratings	★★★★★ (4)	Last Updated	6/6/2012
Downloaded	2,247 times	License	MS-LPL
Favorites	Add To Favorites	Share	
Requires	Visual Studio 2012 RC		
Technologies	Javascript, HTML5, Metro style app		
Topics	User Interface, HTML5, Application Bar, WinJS, Animations, Metro style app		

Report Abuse to Microsoft

Introduction

This example app shows you how to create a stocks app. Key aspects of the app include:

- Fluid layouts
- Metro style controls like ListView

Figure 1-6

Viewing a sample app’s Web page

5. Click the **Browse Code** link, click **StocksSample** in the left pane, click **html**, and then click **dashboard.html**. The HTML markup displays. Scroll through the markup to get a feel for the type of code you'll see many times in this book.

6. Click **css** in the left pane and then click **dashboard.css**. The CSS code displays.
7. Click **js** in the left pane and then click **default.js**. The JavaScript code displays.
8. If you have an app development tool (like Visual Studio) already installed, feel free to download and open the sample app to browse all of the files in the package.
9. You can also go to the HTML5Rocks.com Web site, click the **Posts & Tutorials** menu at the top of the screen, check the **Samples** check box, and then browse how each sample works along with its code.
10. Close any open windows.

MORE INFORMATION

For more information about the app package, visit the "App packages and deployment" Web page at http://bit.ly/H9rsFz.

Understanding Credentials and Permission Sets

The .NET Framework provides a secure environment in which HTML5/JavaScript apps can run. The framework uses security transparency to separate different kinds of code while running, and uses permission sets and identity permissions to control the environment.

Code security is a priority with app developers. The monetary loss from viruses, Trojans, cross-site scripting attacks, and other malware distributed across the Internet increases each year. Creating a safe and secure environment for apps to run in is vitally important to most individuals and organizations today.

1.1

The good news is that the .NET Framework 4.0 supports building and running Metro style apps, among other technologies. The .NET Framework is a Windows component that runs in the background, providing the code-execution environment for scripted or interpreted code (like JavaScript), helping them run with relatively few problems. It also provides an object oriented programming environment for object code.

The .NET Framework now relies more heavily on security "transparency" than in past versions. Transparency prevents application code from running with infrastructure code. The .NET Framework uses permission sets and identity permission. ***Permission sets*** are groups of permissions. Transparent code executes commands that don't exceed the limitations of a permission set, and transparent code is even more limited when it comes to critical code.

The .NET Framework defines several levels of permission sets, which range from Nothing (no permissions exist and code cannot run) to Full Trust (code can access all resources fully).

Identity permissions protect assemblies (compiled code libraries) based on evidence, which is information about the assembly. Each identity permission represents a particular kind of evidence, or credentials, that an assembly must have in order to run.

THE BOTTOM LINE

A session state is created when a user first requests access to an application, and it ends when the session closes, such as when a user logs off. An application state exists from the time a Web browser requests a Web page until the browser closes. Persistent state information is data that exists after a session ends. In HTML5, developers can use the **localStorage** and **sessionStorage** JavaScript methods to deal efficiently with state data. In addition, AppCache enables a user to load data ordinarily stored on a server even when the user is offline.

1.2

1.2

You'll learn more about using AppCache with JavaScript in Lesson 8.

1.2

State management is the process of maintaining Web page information during multiple requests for the same or different Web page. When a user first requests access to an application, the ***session state*** is created. The state ends when the user closes the session.

An alternative to the session state is the application state. The ***application state*** is created when the Web browser sends the first request for a Web page to the Web server, and it ends when the user closes the browser.

1.2

Persistent state information is data that an application needs after the session ends. Many Web applications need to store data (make it persistent) so that users can pick up where they left off when they return to the site.

Storing State Data Using Local and Session Storage

Hypertext Transport Protocol (HTTP) is the protocol that transfers data on the World Wide Web. It defines the actions Web servers and browsers take in response to commands by users. For example, when you enter a uniform resource locator (URL) in the address field in a browser, the browser sends an HTTP command to the Web server requesting the Web page. HTTP is a stateless protocol, which means it doesn't retain data from session to session. When you close a Web browser after visiting a Web site, the data is not saved.

To work around the limitations of HTTP protocol, developers historically have used ***cookies***, which are small files that contain information about the user and the Web site visited and are saved on the user's computer. When a user returns to a visited site, the browser sends the cookies back to the Web server. Cookies help a Web server "remember" a user and customize the user's experience on that site.

However, cookies have proven to be a security risk. In addition, if large amounts of data are involved, all of the data gets sent between the browser and server upon every request, which would cause a noticeable performance decrease to the user. In HTML5, developers can use the Web storage instead, which offers more flexibility, larger data sets, and better performance.

The `localStorage` method allows users to save larger amounts of data from session to session (persistent data), and there's no time limit as to how long the data exists. The `sessionStorage` method keeps data only for one session (until the browser is closed), which is also referred to as "per-tab storage."

Using these methods, specific data is transferred only when requested, so it's possible to store a relatively large amount of data without slowing down the connection or site.

APPCACHE FOR OFFLINE FILES

Another way to use Web storage is to store data locally when a user is offline. The Application Cache, or ***AppCache***, stores resources like images, HTML pages, CSS files,and JavaScript— data that would ordinarily be stored on a server. Because the resources are stored on the cli ent's hard disk or device, the resources load faster when requested.

Using AppCache, a developer uses a text file called a "cache manifest" to specify the files a Web browser should cache offline. Even if a user presses the Refresh button offline, the app will load and work correctly. A cache manifest file looks similar to the following:

```
index.html

stylesheet.css

images/dot.png

scripts/main.js
```

MORE INFORMATION

For more information about state management, and local and session storage, see the "Storing and retrieving state efficiently" Web page at http://bit.ly/H9wH3u.

■

On a touch-screen device, a finger move is called a gesture, and the response by the app to that gesture is called an event. Developing touch-enabled apps requires thorough

THE BOTTOM LINE

1.3

X

You'll learn how to
create JavaScript code for touch interfaces in Lesson 9.

1.3

knowledge of how fingers interact with the screen and planning for different sizes of fingers. You can use JavaScript to create touch-enabled apps, primarily using the touchstart, touchend, and touchmove events.

Today's mobile devices and many PC monitors incorporate touch-screen technology, which makes it easier for many users to interact with the devices and their programs. A simple finger tap selects an object or presses a button, a finger swipe scrolls a list of photos on the screen, and a pinch zooms out on an image.

Any finger move is referred to as a ***gesture***, which can involve a single finger (one-touch, such as press, tap, press and hold, slide to pan, and so on) or a finger and a thumb (two-touch, such as a pinch and stretch or a turn to rotate). The action the application takes in response to a gesture is called a ***touch event***. You can use JavaScript to create touch events in touch enabled apps. In JavaScript, the three primary touch events are touchstart, touchend, and touchmove.

When designing apps for a touch-screen environment, gesture responsiveness is key. Slow performance will frustrate most users. Incorporate physics effects such as acceleration and inertia to create a more fluid interaction between the user and screen.

Visual feedback for successful interactions and other notifications is highly important. This allows the user to understand whether he or she is using the touch landscape appropriately. Snap points help users stop at a location within the interface where intended, even if a gesture is a little off the mark.

You should also keep in mind that users have different size fingers, and it's a best practice to design for wider rather than narrower digits. And of course, users will be either right- or left-handed, so a well-designed app uses vertically symmetric navigation and provides for flipping the screen 90 degrees to go from portrait to landscape or vice versa.

Multi-touch occurs when a user must press multiple buttons or locations at once. This is common with games on a touch-screen device, where the user often uses several fingers and both thumbs simultaneously or in very rapid succession. In this situation, swipes and gestures don't work well, resulting in unintended zooming and scrolling instead. The fix is to disable zooming and scrolling in JavaScript.

Another item to test for in a multi-touch app is the reaction to touch events. There will be many events occurring at the same time, which requires proper tracking of fingers and render ing in a loop to get the best performance.

When developing any touch-enabled app, be sure to test for the following: • Overall responsiveness and fluidity

- Tapping, pinching, rotating, and other common gestures
- Controlled scrolling
- Controlled panning

- Ability to disabled scrolling and panning
- Accuracy of snap points
- Unintended zooming or scrolling, especially in a multi-touch environment • Proper touch event reaction, especially in a multi-touch environment

Designing and developing well-formed touch-enabled apps takes practice, and a lot of testing. If you don't have a touch-screen device, you can use MouseTouch events and a touch-screen emulator or simulator. Try Microsoft Surface SDK and Runtime for Windows 7, or the Windows Simulator tool in Visual Studio 11. A ***touch-screen simulator*** or ***emulator*** imitates a system that only has touch capabilities. Several free emulators are available online.

LEARN ABOUT GESTURES AND TOUCH-ENABLED APPS

GET READY. To learn about different kinds of gestures, perform the following steps: **1.** Go to the Touch interaction design Web page at http://bit.ly/GAJjDL.

2. Read the content on the Web page.

3. Bookmark the page for future reference or locate and click the link that downloads a PDF version of the Web page to your computer.

4. Close the browser window.

MORE INFORMATION

You can find the WC3 touch specifications at http://bit.ly/gBZUjo. For more information about touch and gestures, visit the "Responding to user interaction" Web page at http://bit.ly/H7uO5Q.

Leveraging Existing HTML5 Skills and Content for Slate/Tablet Applications

An advantage for seasoned developers who want to create Metro style apps is that their existing HTML5 skills and code lend themselves well to Metro app development.

A highly flexible aspect of developing touch-enabled Metro style apps is that it doesn't require a big learning curve for developers who are already using HTML5 and other methods of app development. They can apply their existing skills and code to creating Metro style apps almost immediately. A Web developer's experience with HTML, CSS, JavaScript, and JavaScript librar ies is an advantage when transitioning to touch-enabled Metro app development.

In addition, Microsoft.NET Framework and Silverlight developers can apply their XAML, C#, and Visual Basic experience to Metro projects. Game programmers who are well versed with Microsoft DirectX 11 can also apply their skills creating Metro apps.

All apps must be thoroughly tested and debugged to ensure they run reliably and as error free as possible before distribution and deployment. THE BOTTOM LINE

Debugging an application involves detecting, finding, and correcting logical or syntactical errors. A syntax error is a typo in the code or a similar error, which is usually revealed during runtime for interpreted apps. A logic error results in the app behaving differently than expected.

1.3

*

Testing and debugging code is a standard part of app development, and the majority of tools like Visual Studio have debugging features built in to the software. Some errors are easy to detect and fix, whereas others can require hours or even days to resolve, depending on the complexity of the application.

Either way, the testing and debugging phase is highly important for several reasons: • Your goal is to provide a reliable, secure, and useful app to customers. Debugging and testing help to ensure all three are met.

• High-quality apps garner high ratings, which can boost your profits and drive sales of future apps.

• If you plan to publish your app through the Windows Store or another reputable online app marketplace, the store will require validation or certification that your app has been tested.

Validating HTML5 Code

One of the first steps in the debugging and testing phase is to validate your HTML5 code. Validation means verifying the validity of your code. A ***validator*** looks for anything that could cause the code to be interpreted incorrectly, such as missing or unclosed tags, an improper DOCTYPE declaration, a trailing slash, deprecated code, and so on. (Don't worry about those details right now. You'll learn about them in Lesson 2.)

The W3C provides a code validation service for all active versions of HTML on its Markup Validation Service Web page at http://validator.w3.org/. The service is free for anyone to use. You simply click a link to upload your file to the service, or copy and paste the content of your file into a text box on the Web site. After that, click the Check button. The validation service checks your code and reports any errors or problems you need to fix.

A validator is not the same as an emulator or a simulator. A validator actually tests the code and reports inaccuracies, giving you an opportunity to make changes. Emulators and simula tors simply provide an environment in which to run code.

The W3C also provides a link checker at http://validator.w3.org/checklink. This service checks that all links in your HTML file are valid. The CSS Validation Service at http://jigsaw.w3.org/css-validator/ checks your CSS files.

Validating a Package

Microsoft provides a free tool called the Windows App Certification Kit for testing local apps. The kit is a type of validator that tests your app on your computer before you attempt to package and publish it to the Windows Store.

The Windows App Certification Kit is included in the Windows Software Development Kit (SDK) for Metro style apps, available on the Microsoft Web site. To use the kit, you must first package and install the app locally using an app development tool. Then open the kit, select the application you want to validate, and run the validator. A report displays noting any problems with the app. The Windows App Certification Kit might also be available as a menu choice within your app development tool.

Correct the problems in an app development tool and then test the application again. You'll repeat this process until your app validates.

USE THE W3C MARKUP VALIDATION SERVICE

GET READY. To become familiar with the W3C Markup Validation Service, perform the following steps:

1. Go to the W3C Markup Validation Service Web page at http://validator.w3.org/.
2. Click the **Validate by File Upload** tab.
3. Click **Browse**.
4. Navigate to and select an HTML file from one of your sample apps. Click **Open**, and then click **Check**.
5. Scroll down the page and read the errors and warnings, if any. Figure 1-7 shows an example.

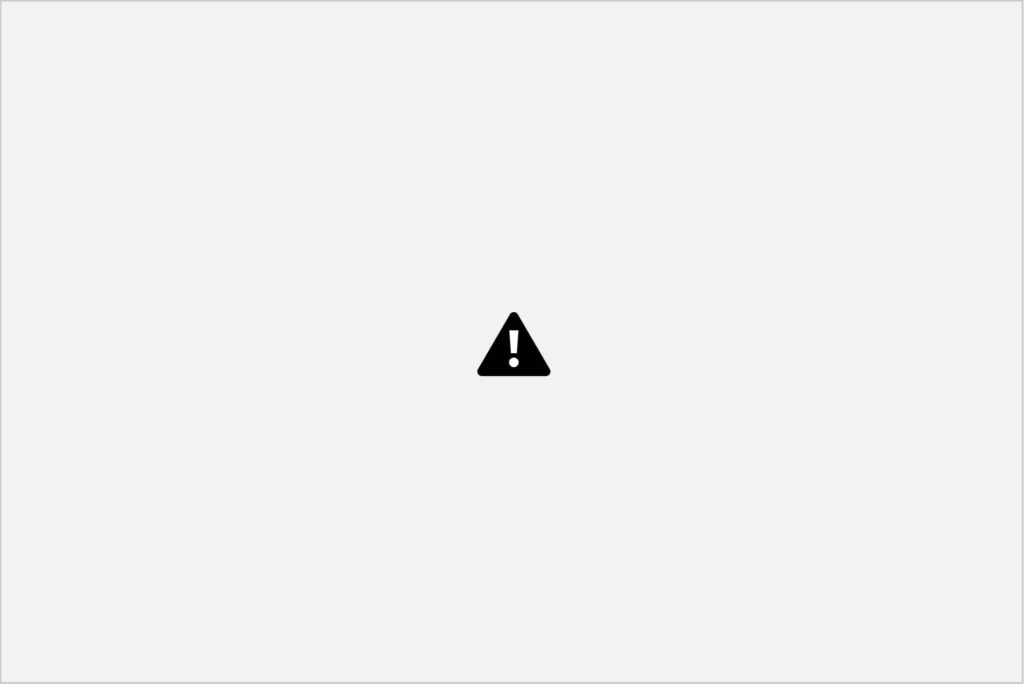


Figure 1-7

Errors and warnings as a result of attempting to validate an HTML Web page

6. If the validator provides links to more information about errors or warnings, click through to at least two of them and read the information.
7. When you're finished, leave the Web browser open.

USE THE W3C CSS VALIDATION SERVICE

GET READY. To become familiar with the W3C CSS Validation Service, perform the following steps:

1. Go to the W3C CSS Validation Service Web page at http://jigsaw.w3.org/css-validator/ (see Figure 1-8).
2. Click the **By file upload** tab.
3. Click **Browse**.
4. Navigate to and select a CSS file from one of your sample apps. Click **Open**, and then click **Check**.

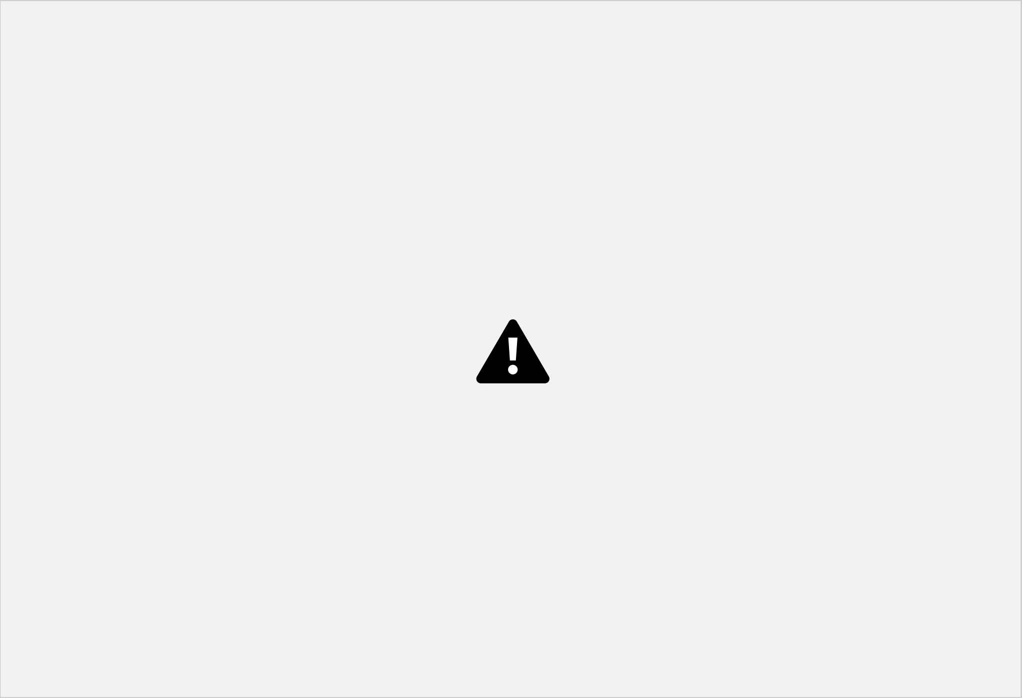


Figure 1-8

The W3C CSS Validation Service Web page

5. Scroll down the page and read the errors and warnings, if any.
6. If the validator provides links to more information about errors or warnings, click through to at least two of them and read the information.
7. When you're finished, leave the Web browser open.

■

THE BOTTOM LINE

1.4

Once your app has been tested, debugged, and the code validated or certified, you need to take a few more steps to prepare it for upload to a marketplace such as the Windows Store. You can use Visual Studio 12 or Visual Studio 12 Express to complete the project.

Publishing your app to a public marketplace like the Windows Store is the pinnacle of all of your planning, designing, coding, and testing. The ***Windows Store*** is an online global marketplace for Metro style apps. Publishing your app for distribution through the store can possibly turn a good idea into a lucrative venture.

Another bonus to selling through the Windows Store is that you get access to several handy tools, such as Microsoft Visual Studio Express and Microsoft Expression Blend. You can also download personalized app telemetry data, which can greatly speed up app creation and deployment.

Before publishing your app to the Windows Store, you must do the following: • Sign up and pay for a Windows Store developer account, and reserve a name for your app. You'll also need to edit your app's manifest file.

- Go through the app submission checklist at http://bit.ly/HAPmbk. The checklist includes tasks such as naming your app, choosing selling details such as selecting appropriate pricing and a release date, assigning an age rating, describing your app, and more.

*

You must sign up and pay for a Windows Store developer account to add your app to the store menu.

- Use the Windows App Certification Kit to test your app, if you haven't done so already.
- Capture some screen shots of significant or unique features of your app to showcase in the store. You can use the Snipping Tool, which is built into Windows 7 and Windows 8, to capture screen shots or you can use another tool of your choice.
- Have other testers or developers test your app on as many different devices and platforms as possible, especially if you tested it only in a simulator or emulator.
- Include a privacy statement if your app gathers personal information or uses copyrighted software to run.

When you're ready, use your app development tool (such as Visual Studio 12 or Visual Studio 12 Express for Windows 8) to create a final app package and then upload it to the Windows Store.

It's customary to wait for approval from the store. If approved, your app will be certified and listed. However, even after all of your preparatory work, your app could be rejected, which means you must fix any problems noted by app store personnel if you want to retest and republish the app.

BECOME FAMILIAR WITH THE WINDOWS STORE

MARKETPLACE **GET READY**. To learn more about Windows Store requirements, perform the following steps: 1. Open Internet Explorer, then go to www.bing.com.

2. Search for Windows Store Marketplace and go to the site.

3. Browse the categories of apps. Note the three highest rated apps, and another three apps that interest you.

4. Read the description of each app and make notes that could help you write an appealing description for your app.

5. Note the number and quality of screen shots provided for those apps. 6. Note the quality of the launcher icon and any other graphical details. 7. Note the price and age rating of each app.

8. Note any other details that might help you sell your app when it's ready. 9. When you're done, close all open windows.

SKILL SUMMARY

- HTML5 is the latest HTML standard and a family of technologies that includes HTML, CSS, and JavaScript. Although the HTML5 standard won't be finalized for a few years, most modern Web browsers already support HTML5 elements, and HTML5 app development for Web and mobile device browsers is well underway.
- The HTML5 family includes many new markup tags and technologies like media queries, geolocation, Modernizr, and much more.
- The general steps for creating an app are: plan the project, design a UI, update the app mani fest, write code, build the app, debug and test the app, package the app, and deploy the app.
- The Windows Runtime (WinRT) environment is the foundation of the Windows 8 operating system and provides functionality to Metro style apps.
- Metro style apps created with JavaScript and that are opened in Internet Explorer are run by the WWAHost.exe process. This is a different process than the host process that ordi narily runs Internet Explorer.
- The purpose of an app package is for ease of distribution and deployment. Application packaging bundles an app's files and folders into an app package.
- The .NET Framework provides a secure environment in which HTML5/JavaScript apps can run. The framework uses security transparency to separate different kinds of code while

running, and uses permission sets and identity permissions to control the environment.

- A session state is created when a user first requests access to an application, and it ends when the session closes.
- An application state exists from the time a Web browser requests a Web page until the browser closes.
- Persist state information is data that exists after a session ends.
- In HTML5, developers can use the localStorage and sessionStorage JavaScript methods to deal efficiently with state data.
- AppCache is a type of Web storage that enables a user to load data that's ordinarily stored on a server even when the user is offline.
- On a touch-screen device, a finger move is called a gesture, and the response by the app to that gesture is called an event.
- Developing touch-enabled apps requires thorough knowledge of how fingers interact with the screen and planning for different sizes of fingers.
- You can use JavaScript to create touch-enabled apps, primarily using the touchstart, touchend, and touchmove events.
- An advantage for seasoned developers who want to create Metro style apps is that their existing HTML5 skills and code lend themselves well to Metro app development.
- All apps must be thoroughly tested and debugged to ensure they run reliably and as error free as possible before distribution and deployment.
- Once your app has been tested, debugged, and the code validated or certified, you need to take a few more steps to prepare it for upload to a marketplace such as the Windows Store. You can use Visual Studio 11 or Visual Studio 11 Express to complete the project.

Fill in the Blank

Complete the following sentences by writing the correct word or words in the blanks provided.

1. HTML is a __________ language, not a programming language, which means HTML uses markup tags such as <body> and <h1> to describe parts of a Web page.

2. ____________________ defines styles for HTML in a separate file, so you can easily change fonts, font sizes, and other attributes.

3. Windows 8 users the __________ user interface (UI).

4. The __________ is the foundation of the Windows 8 operating system, and is made up of layers that provide functionality to Metro style apps and the Windows shell.

5. __________ ___ is the process of bundling an application and its various files into an app container, making it easy to distribute and deploy the app. The app package is the result of this process.

6. The ____________ state is created when the Web browser sends the first request for a Web page to the Web server and it ends when the user closes the browser.

7. The ___________ method keeps data only for one session (until the browser is closed), which is also referred to as "per-tab storage."

8. Any finger move is referred to as a ________, which can involve a single finger (one-touch) or a finger and a thumb (two-touch).

9. A ________ looks for anything that could cause code to be interpreted incorrectly, such as missing or unclosed tags, an improper DOCTYPE declaration, a trailing slash, deprecated code, and so on.

10. The ____________ is an online global marketplace for Metro style apps.

Multiple Choice

Circle the letter that corresponds to the best answer.

1. Which three components are the primary elements of the HTML5 family? **a.** XML
b. HTML
c. CSS
d. JavaScript

2. JavaScript is a type of:
a. Program compiler
b. Markup language
c. Scripting language
d. Validator

3. All of the following are true of HTML5 except:
a. It requires Windows 8
b. It can be used to create Web apps and PC and device apps
c. It is platform-independent
d. It is built on an open standard

4. Which operating system environment allows a developer to access a camera or webcam?
a. localStorage
b. WinRT
c. the session state
d. Metro

5. You are developing a Metro style app and want the app to access another app. Where do you declare the interaction?
a. App manifest
b. CSS
c. At the top of the HTML file
d. Nowhere; you do not have to declare the interaction

6. Which of the following is used to create an app package?
a. JavaScript
b. CSS
c. DOM
d. An app development tool

7. Which API allows programs and scripts to update content, structure, and styles on the fly? **a.** JavaScript
b. WinRT
c. The DOM
d. RTE

8. AppCache, localStorage, and sessionStorage are forms of:
a. Web storage
b. HTML commands
c. Standards
d. Namespaces

9. Which of the following does not usually work well with multi-touch environments and

should be disabled? (Choose two.)
a. Tracking
b. Zooming
c. Scrolling
d. Gesturing

10. Which tool is a type of validator that tests your app on your computer before you attempt to package and publish it to the Windows Store?
a. WinRT
b. Windows 8
c. W3C Markup Validation Service
d. Windows App Certification Kit

True / False

Circle T if the statement is true or F if the statement is false.

T F 1. An application programming interface (API) is a list of instructions letting a pro gram communicate with another program.

T F 2. A best practice is to publish your app without validation to perform live online testing.

T F 3. An emulator searches HTML and CSS documents, looking for errors.

T F 4. It's a best practice to design touch-enabled apps for wider rather than narrower digits.

T F 5. A platform-independent app can run on different desktop and mobile device oper ating systems.

Scenario 1-1: Understanding New Features in the HTML5 Family

Your manager, Marylyne, wants to learn about the HTML5 family to decide if the company should begin using it on new projects. She asks you to provide her a list of five or six new features. What items do you include in the list?

Scenario 1-2: Creating an App

Marylyne approaches you again, this time wanting to know what is involved in creating an HTML5 app. She asks you to provide an outline. What steps do you include in the outline?

Scenario 1-3: Sharing Touch-Enabled App Development Tips

Antoine is working on a touch-enabled app and asks you for development tips and items he should be sure to test on his tablet. What do you tell him?

Scenario 1-4: Publishing an App to the Windows Store

Sammy created his first app and wants to publish it to the Windows Store. What are three preparatory steps he should take?

Building the User Interface by Using HTML5: Text, Graphics, and Media

EXAM OBJECTIVE

MATRIX

LESSON

S /CONCEPTS MTA EXAM OBJECTIVE MTA EXAM OBJECTIVE N

Understanding the Essentials
of HTML

Choosing and Configuring HTML5 Choose and configure HTML5 tags 2.1
Tags to Display Text Content to display text content.

Choosing and Configuring HTML5 Choose and configure HTML5 tags 2.2
Tags to Display Graphics to display graphics.

Choosing and Configuring HTML5 Choose and configure HTML5 tags 2.3
Tags to Play Media to play media.

attribute

audio element

canvas element codec

compression

deprecation

doctype

element

empty tag

entity

figcaption element

figure element

global attribute

nesting

raster image

render

Scalable Vector Graphics (SVG) tags

valid

vector image

video compression

video element

The busy Web site developers at Malted Milk Media have asked you to research new markup that's available in HTML5. They're particularly interested in graphics and multimedia-related tags. Your task is to learn all that you can about new HTML5 markup tags and prepare brief descriptions and provide examples of each.

Hypertext Markup Language (HTML) uses markup to describe content for display on a Web page. An element is the combination of tags and the content they enclose. You may

THE BOTTOM LINE

need to use special characters on a Web page, which requires character encoding. Finally, every Web page requires the `doctype` declaration at the top of the page.

Hypertext Markup Language (HTML) is called a markup language because you use it to describe (mark up) pieces of content to display on a Web page. A Web page with markup means it includes ***tags***, which are keywords that help to give an HTML page structure. (You'll learn more about tags shortly.) The key to using HTML is learning which tags to use and when. The combination of content, tags, and perhaps graphics, multimedia, and so on are what build a Web page.

You can easily identify an HTML document because it has an .htm or .html file extension. When a Web browser or mobile device such as a smartphone opens an HTML file, it ***renders*** (interprets and reproduces) the content of the page.

Basic Markup and Page Structure

Every HTML page includes tags. A tag is a keyword surrounded by angled brackets. Most tags come in pairs; one tag is called the opening or start tag, and the other is the closing or end tag. A tag pair is case sensitive—a closing tag must have the same case as the opening tag. A closing tag is identical to an opening tag except the closing tag includes a slash before the keyword.

Tags surround content and give it definition. For example, this markup creates a first-level heading:

```
<h1>Pet Care 101</h1>
```

HTML also uses some single tags, like `<br />` for a line break and `<hr />` for a horizontal line. In HTML 4, these tags are called ***empty tags*** because they don't require an end tag. HTML5 is less restrictive than HTML 4. You don't have to include end tags for all elements (although some elements still require start and end tags), and you can enter tags in uppercase or lowercase. However, this book uses start and end tags, and all lowercase for markup, for consistency.

There are many tags available for HTML pages. Some of

the most commonly used tags are listed in Table 2-1. <body>—are required on every Web page.
The first four—<html>, <head>, <title>, and

Table 2-1

Common HTML tags TAG D

<html>	Identifies the page as an HTML document. The <html> tag encompasses everything on the page other than the doctype declaration at the top.
<head>	Contains markup and code used by the browser, such as scripts that add interactivity, and keywords to help search engines find the page. Content in the <head> tag can also include formatting styles for the page.
<title>	Displays the title of the Web page, which appears at the top of the Web browser, usually on the page's tab in a tabbed browser.
<body>	Surrounds content that's visible on the Web page when viewed in a Web browser.
<a href=URL>	Generally used to anchor a URL to text or an image; can also create a named anchor within a document to allow for linking to sections of the document.
<b>	Applies boldface to text.
<hx>	Creates a heading, which can be first level (h1) through sixth level (h6).
<img>	Inserts an image from a file or another Web site.
<p>	Defines text as a paragraph.

A tag pair or an empty tag is also called an ***element***. An element can describe content, insert graphics, and create hyperlinks.

USING ATTRIBUTES

Not all tags describe data on their own or at least not in enough detail for rendering, so some elements must include ***attributes***, which are modifiers of HTML elements that provide addi tional information.

Attributes are easy to use and are just extensions of elements. You add attributes to elements according to this basic syntax:

```
<tag attribute="value">
```

Notice that the attribute and its value are both inside a tag. You must include an attribute within a tag so that the Web browser knows how to handle the attribute. A good example of an attribute is when creating a hyperlink, as follows:

```
<a href="http://www.example.com">This is a link.</a>
```

The Web browser uses the combination of the anchor element and the href attribute to display a hyperlink. Figure 2-1 shows how a Web browser interprets this bit of markup.

*

A good Web page editor or app development tool should show you which attributes you can use with an element, which is a time saver. The tool should also help you debug the markup if you used an attribute incorrectly.

Figure 2-1

A hyperlink is the result of the anchor element using the href attribute

Figure 2-2

Incorrectly nesting tags

Two of the most common uses of attributes are to create hyperlinks and to insert simple graphics. You'll learn how to work with graphics later in this lesson. HTML5 includes several ***global attributes***, which you can use with any HTML5 element. Examples of global attri butes include `id`, `lang`, and `class`, among many others.

NESTING ELEMENTS

How a Web browser displays your HTML depends on the way you combine elements, their attributes (if any), and content. When two or more elements apply to the same block of text, you should nest tag pairs appropriately so that they do what you intended. ***Nesting*** means to place one element inside another. Here's an example of correct nesting:

```
<p>Make sure your pet has plenty of <i><b>fresh
water</b></i>  during hot weather.</p>
```

In this case, we want the words "fresh water" to stand out so they are italicized and bolded by using the `<i>` and `<b>` tags. If you placed the `</b>` end tag after the `</p>` end tag (shown below), the words "fresh water during hot weather" would appear bold but only "fresh water" would be italicized. It would look awkward, as shown in Figure 2-2.

```
<p>Make sure your pet has plenty of <i><b>fresh
water</i>  during hot weather.</p></b>
```

The rule for nesting is that nested tags must be closed before their parent tags. Looking back at the correct example, notice that the paragraph element opens first, followed by the italic element, and then the font element. Then the bold element closes, followed by the italic ele ment, and finally the paragraph element. The italic and bold elements are completely nested within the paragraph element.

UNDERSTANDING ENTITIES

An ***entity*** is a special character, such as the dollar symbol, the registered trademark (a capital R within a circle), and accented letters. The process of incorporating entities in a Web page is called character encoding. Today's Web editing tools and browsers do a good job of handling special characters that appear on your keyboard, such as those above the number keys. In most cases, those characters render without any problems.

With some browsers, the character you expected doesn’t appear and you get a gibberish character or symbol instead. Those situations are easy to handle. Each special character that can be reproduced in a Web page has an entity name and a numerical code. You can use either in a Web page. However, it’s generally safer to represent symbols like the trademark using a numbered entity to ensure proper rendering in a wide variety of browsers.

An entity begins with an ampersand (&) and ends with a semicolon (;). For example, the entity &reg; represents the registered trademark symbol, and its numerical code is &#174; . When a browser encounters an ampersand, it tries to match the characters that follow with an entity. If the browser finds a match, it displays the special character in place of the entity. Table 2-2 lists a few commonly used entities.

Table 2-2

A sampling of entities for HTML5

SYMBOL DESCRIPTION ENTITY NAME C © Copyright

&copy; &#169; ° Degree &deg; &#176; $ Dollar &dollar;

&#36; % Percent &percnt; &#37; ® Registered

trademark &reg; &#174;

Another important thing to know about character encoding in HTML5 is that you should use UTF-8 encoding whenever possible, because most browsers use UTF-8. That means you add the following declaration to the head element:

```
<meta charset="UTF-8">
```

The HTML5 specification requires that the whole meta element fits in the first 1,024 bytes of the document, which is why you include it at the top of the page in the head element.

✚ MORE INFORMATION

For a list of entities supported in HTML5, go to http://dev.w3.org/html5/html-author/charref.

UNDERSTANDING THE DOCTYPE

The ***doctype*** is a declaration that is found at the very top of almost every HTML document. When a Web browser reads a doctype declaration, the browser assumes that everything on the Web page uses the language or rules specified in the declaration.

In HTML 4, all <!DOCTYPE> declarations require a reference to a DTD, which stands for Document Type Definition. The DTD is simply a set of rules that help a Web browser turn tags and content into the pages you see on the Web. There are a few different DTDs that an HTML 4 Web page can use. Because of how HTML5 was created, it doesn’t require a reference to a DTD.

In HTML 4, the doctype declaration specifies the HTML page’s language and DTD, and looks quite complex. Here’s an example:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
 "http://www.example.com/TR/xhtml11/DTD/
 xhtml11.dtd">
```

The new HTML5 doctype, in comparison, is very simple:

```
<!doctype html>
```

The HTML5 doctype is case-insensitive, so the keyword “doctype” can be uppercase or lowercase. This simplified doctype is partially responsible for why HTML5 pages easily lend themselves for viewing in a Web browser on a computer or a mobile device. HTML5 is designed to be broadly compatible with both new and old Web browsers, and the mobile device environment.

EXPLORING THE MARKUP OF A SIMPLE WEB PAGE

An example of markup and content for a simple HTML5 Web page looks like this:

```
<!doctype html>

<html>
 <head>
 <title>78704 Pet Services</title>
 </head>

<body>
 <p>Your dog is a friend for life. Why not
 provide the best care possible?</p>
</body>

</html>
```

The blank lines between parts of the page, such as between the doctype declaration and the <html> tag, don't appear on a Web page. Neither do indents, such as those for the paragraphs. (Notice that the paragraph elements are indented a bit from the <body> tags. Blank lines and indents simply help you read the markup more easily in an editing tool.

Figure 2-3 shows the rendered Web page for the previous HTML markup.

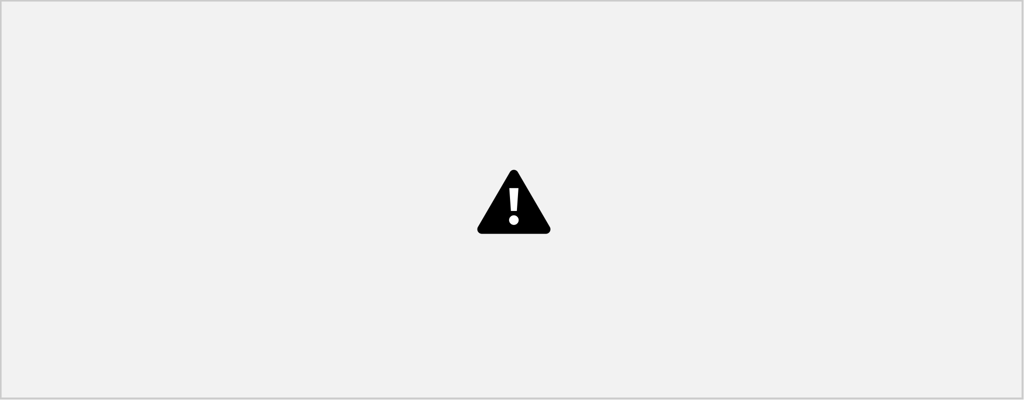


Figure 2-3

A simple Web page rendered by a browser

Recall from Lesson 1 that you can use the W3C's validation service at http://validator.w3.org to check and validate HTML code. If a Web page adheres to the specifications perfectly, it is considered ***valid***.

CREATE A SIMPLE WEB PAGE

GET READY. To create a simple Web page and see the effect of missing tags, nesting, and enti ties, perform the following steps:

1. On your computer or a flash drive, create a subfolder within the My Documents folder that will hold the files you work on in lessons throughout this book. This is your working folder. You can name the subfolder **HTML5** or something similar.

*

Figure 2-4

2. Open a Web page editor, app development tool, or even a simple text editor like Notepad and type the following:

```
<!doctype html>
<html>
 <head>
 <title>78704 Pet Services</title>
 </head>
<body>
<h1>Care and Feeding</h1>
 <p>Your dog is a friend for life. Why not provide the
best  care possible?</p>
 <p>Make sure your pet has plenty of <i><b>fresh
water</b></i>   during hot weather. When taking your
dog on long walks,   bring along a collapsible water dish
and bottled water.   You can find specialty water dishes
at many pet supply  stores for $10 or less.</p>
</body>
</html>
```

You have a lot of choices when it comes to editors and development tools. Notepad is the built-in text editor in Windows, but you can download Notepad++ for free from the Web. Notepad+++ offers features that make it easier to create and edit HTML documents. TextWrangler has a similar feature set and is designed for Macintosh systems. Free HTML editors include HTML-Kit and KompoZer. Development tools include Microsoft Visual Studio, Visual Studio for Web, Microsoft Web Matrix, and Microsoft Expression Web, among many others. All of these applications enable you to create and edit HTML files.

3. Save the file as **L2-pet-orig.html** in the working folder you created in My Documents. 4. Navigate to your working folder and open the HTML page in a Web browser. It should look similar to Figure 2-4.

The 78704 Pet Services Care and Feeding Web page

*

5. To see the effect of a missing tag in a tag pair, delete the </b> end tag after "water." Create a new file to test the changes by saving it as **L2-pet-test.html** and open it in the browser. Now all of the content from "fresh

water” to the end of the document is in boldface.

In Internet Explorer 9, you can press **F12** to open browser mode. This mode enables you to edit pages without leaving the browser. In addition, you can click **Document Mode** on the menu bar and then select an older version of the browser to see how a page renders.

6. To see the effect of improper nesting, move the </i> end tag to appear after the last </p> tag. Save **L2-pet-test.html** again and view it in a browser. Now all of the content from “fresh water” to the end of the document is in boldface and italics, as shown in Figure 2-5.

Figure 2-5

Effects of improper nesting of tags

7. Close the **L2-pet-test.html** file in the editor and open **L2-pet-orig.html**. 8. Add a copyright line to the bottom of the page by pressing **Enter** a few times after the closing </p> tag and typing `<p>&copy; 2012</p>`. Substitute the current year for “2012”, if necessary. Press **Enter** to add a blank line. Make sure the copyright line is above the </body> and </html> end tags.

9. Create a new file again by saving L2-pet-test.html as **L2-pet-copyright.html** and view it in the browser. Does the circle C symbol appear as shown in Figure 2-6? If not, change &copy; to **&#169;**, save the file, and then view it again.

Figure 2-6

A copyright symbol appears in the lower-left corner

Copyright line with symbol

*

When viewing Web pages that you’re editing, it’s best to use a variety of Web browsers to ensure your markup renders as expected for the widest audience. Some editing tools let you select a browser for previewing Web pages from a list. If your

tool doesn't include that option, you'll need to install three or one.
four different browsers and open your Web pages in each

10. Go to the W3C Markup Validation Service Web page at **http://validator.w3.org**. Upload **L2-pet-copyright.html** and click **Check** to have the service check it. Fix any errors reported by the checker that relate to missing tags or typos, if any.

11. You probably received an error message about character encoding. To fix this, open **L2-pet-copyright.html** in your editing tool, insert `<meta charset="UTF-8">` in the head element, on its own line, just before the title.

```
<head>
<meta charset="UTF-8">
            <title>78704 Pet Services</title>
</head>
```

12. Save the file, upload it to the validation checker again, and check it. The checker should indicate that your file is valid.

13. Leave the editing tool and Web browser open if you're continuing immediately to the next section.

MORE INFORMATION

If you find yourself struggling with the topics in this section, consider taking some tutorials such as those at the W3Schools.com Web site.

THE BOTTOM LINE

HTML5 uses most of the same elements and attributes specified in HTML 4, and has introduced some new tags, modified the preferred usage of others, and no longer supports certain elements. New text-related elements include command, mark, time, meter, and progress. A few of the deprecated elements are basefont, center, font, and strike.

All of the elements covered in the first section in this lesson work well in HTML5, even though they have been used for years in previous versions of HTML. For the most part, HTML5 replaces very little HTML syntax. That means developers can still use most of the same elements they always have. Some elements have the same tag but slightly tweaked func tionality, which you'll learn about shortly.

HTML5 also includes many new elements that increase the functionality of Web pages or streamline the markup. These include multimedia elements such as audio and video, and elements that make the structure of a Web page seem more intuitive. Structure-related tags include elements for page sections, headers, footers, navigation, and even sidebars. If you create Web forms, new form features make creation and validation much easier. This section, however, focuses on HTML5 markup for text.

HTML5 layout, section ing, and form creation markup is covered in Lesson 3.

Text Elements from HTML 4 with New Meaning or Functionality

Some HTML 4 text-related elements now have slightly different meaning or functional ity in HTML5. The elements include <b>, <i>, <strong>, <em>, and <small>.

The <b> element should now be used to offset text without conveying importance, such as for keywords or product names. The <i> element now indicates content in an alternate voice or mood, like spoken text. The <strong> element indicates strong importance, whereas the <em> element indicates emphatic stress. The <small> element should be used for small print, like a copyright line.

Let's look at some of the text elements carried over from HTML 4 that have slightly different meaning or functionality in HTML5:

- **<b>:** This commonly used element has always represented boldface, and was often used for emphasis or to convey importance. The W3C suggests you now use it to indicate "stylistically offset" text without conveying importance. Use <b> for keywords, product names, and actionable items (such as items you click or press in a list of how-to steps). For example:

```
<p>Click the <b>Check</b> button, and then click
<b>OK.</b> </p>
```

- **<i>:** The italic element is now used for text in an "alternate voice or mood." This could be spoken text, thoughts, or something similar that doesn't convey importance or emphasis. It may also include technical terms and transliterated foreign words. For example:

```
<p><i>He truly has a kind heart,</i> she thought.
```

- **<strong>:** The strong element is for strong importance, where the content is more important than nearby words. For example:

```
<p>Courtney wore the <strong>same</strong> outfit to work three
days in a row.</p>
```

- **<em>:** The emphasis element indicates emphatic stress. For example:

```
<p>You should <em>always</em> validate your HTML
markup before sharing it with others.</p>
```

- **<small>:** The small element should be used for small print or side comments. This element is useful for copyright lines or adding a source line to an image. For example:

```
<p><small>Copyright 2012 by XYZ
Corporation</small></p>
```

The intended functionality for some of these elements in HTML5 can be confusing, such as knowing when to use the italic element. The best approach is to strive for consistency within a page or Web site, and watch how other developers use the same elements.

MODIFY TEXT-RELATED TAGS IN A WEB PAGE

GET READY. To modify tags in a Web page, perform the following steps:

1. In your editing tool, open **L2-pet-copyright.html** if it's not already open.
2. In the following paragraph, replace the italic and bold tags with the strong element.

```
<p>Make sure your pet has plenty of <i><b>fresh
water</b></i> during hot weather.</p>
```

The resulting markup will look like this:

```
              <p>Make sure your pet has plenty of <strong>fresh
   water</strong> during hot weather.</p>
```

Note that the strong element will look like the bold element. The W3C prefers that you use <strong> over <b>, although they seem to produce nearly identical results.

3. Add **<small>** start and end tags to the copyright line, nesting them properly within the paragraph tags.

4. Save the file as **L2-pet-modified.html** and view it in a Web browser. See Figure 2-7.

Figure 2-7

Using <strong> and <small> tags

Strong element applied to “fresh water”Small element applied to copyright line

5. Leave the editing tool and Web browser open if you’re continuing immediately to the next section.

New Text Elements in HTML5

New text-related elements in HTML5 include <command>, <mark>, and <time>, along with a few others. The <command> element creates a command button. When the user clicks a command button, a command executes. The <mark> element highlights text on a page, similar to the highlighting feature in Microsoft Word. The <time> element dis plays a machine-readable time and date, such as 10:10 A.M., CST, July 19, 2012, which is handy for blogs and calendars, and potentially helps search engines provide better results when time and date are part of the search criteria.

Let’s take a look at some of the new text elements in HTML5 along with some examples:

- **<command>:** The command element is used to define a command button that users click to invoke a command. The command element has many attributes you can use, such as type, label, title, icon, disabled, checked, and radiogroup. For example:

```
<menu label="Music Genre">

 <command type="radio" radiogroup="musicgenre"
label="Art">

 <command type="radio" radiogroup="musicgenre"
```

```
label="Popular">

 <command type="radio" radiogroup="musicgenre"
label="Traditional">

</menu>
```

• **<mark>:** The mark element is very handy for highlighting text on a page. You could use it on a search results page, for example, or to set off a block of text that you want to draw to the reader's attention. For example:

```
<p>Since I started jogging last fall, I have <mark
style="background-color:yellow;">lost 35
pounds</mark>.</p>
```

• **<time>:** The time element indicates content that is a time or date, which can be made machine-readable with the `datetime` attribute. The time element defines time on a 24-hour clock and a date in the Gregorian calendar. One benefit of making times and

dates machine-readable on your Web page is that it helps search engines produce better search results. For example:

`<time datetime="2013">` means the year 2013

`<time datetime="2013-04">` means April 2013

`<time datetime="04-15">` means 15 April (any year)

Two other new elements are meter and progress. The meter element indicates content that's a fraction of a known range, such as disk usage. The progress element indicates the progress of a task towards completion.

USE THE MARK ELEMENT

GET READY. To use the mark element to highlight text, perform the following steps:

1. In your editing tool, open **L2-pet-modified.html** if it's not already open.
2. Modify the following paragraph by inserting the mark element around the text "friend for life".

```
<p>Your dog is a <mark style="background-color:orange;">
friend for life</mark>.
```

3. Create a new file by saving it as **L2-pet-mark.html** and view it in a Web browser. Figure 2-8 shows the highlighted text.

Figure 2-8

The mark element highlights specific text

Highlighted text

4. Leave the editing tool and Web browser open if you're continuing immediately to the next section.

Text Elements Not Used in HTML5

While new elements become available, the W3C earmarks other elements for eventual removal because their functionality is no longer useful. Removing elements from the list of available HTML elements is referred to as ***deprecation***. (The same thing applies to attributes.)

Lesson 4 explores CSS essentials and the sepa ration of presentation (style) from content.

Deprecation may be due to a new element replacing the functionality of an older element, or the preference of a new method of formatting over an older element.An example of the latter is formatting with Cascading Style Sheets (CSS). Using CSS to change the look and feel of text, images, and other Web content separates style from content. The W3C has been nudging developers toward using CSS to control Web page formatting instead of using local formatting for quite some time, and it's clearly the method to be used in HTML5.

This makes sense because you can easily change styles in CSS that apply across a Web page or even a Web site. Inserting individual styles throughout even a single Web page can be time consuming to modify when a change becomes necessary.

The following HTML elements are considered deprecated and are not supported in HTML5 pages:

- **<acronym>:** Defines acronyms in HTML 4 that can be spoken as if they are a single word, such as GUI for graphical user interface. Use the `<abbr>` tag instead. • **<applet>:** Defines an embedded applet. Use the `<object>` tag instead. • **<basefont>:** Defines a default font color, font size, or font family for all the text in a document. Use CSS for applying all fonts.
- **<big>:** Makes text bigger relative to the current font size. Use CSS instead.
- **<center>:** Center-aligns text and content. Use CSS instead.
- **<dir>:** Defines a directory list. Use the `<ul>` tag instead.
- **<font>:** Specifies the font face, font size, and font color of text. Use CSS instead. • **<frame>:** Defines a particular frame (a window) within a frameset (see the next bulleted item).
- **<frameset>:** Defines a frameset for organizing multiple frames (windows).
- **<noframes>:** Displays text for browsers that don't support frames.
- **<strike>:** Defines strikethrough text. Use the `<del>` tag instead for small amounts of text, or use CSS for large blocks of text.
- **<tt>:** Defines teletype or monospaced text. Use the `<code>` tag or CSS instead.

Just because an element isn't supported doesn't mean it won't work within certain browsers. Many users still use older versions of browsers, and many deprecated elements render well in those browsers. However, a best practice is to create pages assuming Web page visitors use a current or near-current browser, which means using the latest HTML elements. If you know all of your Web page visitors use an older browser version, it's acceptable to use deprecated elements. Regardless, if you need to apply a lot of formatting to any Web page, it's best to use CSS for efficiency.

The following attributes are not used in HTML5, although these attributes are not actually part of any HTML specification:

- **bgcolor:** Applies a specified background color to whatever content its associated element describes, which is usually a table or a page. Use the CSS property `background-color` instead.

- **bordercolor:** Applies a specified color to the cell of a table. Use the border-color CSS property instead.
- **bordercolorlight:** Applies a specified color to the upper and left corners of a table cell. Use the border-color CSS property instead.
- **bordercolordark:** Applies a specified color to the lower and right corners of a table cell. Use the border-color CSS property instead.

Like with deprecated elements, you may use these attributes if you know that your Web page visitors use older browsers. Be aware that your attempts to validate your Web page will result in errors, which you can ignore if you're certain your visitors' browsers support the attributes.

MORE INFORMATION

To find out about new features of HTML5, browse the "Learn HTML5 in 5 Minutes!" Web page at http://msdn.microsoft.com/en-us/hh549253 and the W3C "HTML elements" Web page at http://dev.w3.org/html5/markup/elements.html#elements.

SEE THE EFFECTS OF DEPRECATED ELEMENTS

GET READY. To see the effects of deprecated elements in an HTML5 Web page, perform the following steps:

1. In your editing tool, open **L2-pet-mark.html** if it's not already open.
2. Modify the h1 heading to incorporate the center element, as shown:

```
<h1><center>Care and Feeding</center></h1>
```

3. Create a new file by saving it as **L2-pet-temp.html** and view it in a Web browser. Did the element center the heading in your browser?
4. Add the big element to the following content, as shown:

```
            <p>Your dog is a <mark style="background-color:orange;">
<big>friend for life</big></mark>.
```

5. Save the file and view it in a Web browser. Do you see the effect of the big element? See Figure 2-9 as an example.

Figure 2-9

The effects of the center and big elements

6. Go to the W3C Markup Validation Service Web page at **http://validator.w3.org**. Upload **L2-pet-temp.html** and click **Check** to have the service check it.
7. Notice that the validator displays errors regarding use of the deprecated elements. What can you conclude about using deprecated elements in HTML5? (Deprecated elements don't validate but many of them still render properly in a Web browser.)

8. Close **L2-pet-temp.html** and leave the editing tool and Web browser open if you're continuing immediately to the next section.

Use the img element to display linked images in a Web page. The images can be located with the Web pages HTML files, usually in an images subfolder, or on a different server or Web site. The figure and figure caption elements are new to HTML5, and give you more control of the type of image you are displaying and the ability to include captions.

THE BOTTOM LINE

The canvas element is used for drawing, rendering, and manipulating images and graphics dynamically in HTML5. Scalable Vector Graphics (SVG) enables you to create scalable objects that resize to best fit the screen on which they're viewed, whether a PC screen or a smartphone.

2.2

Table 2-3

You can display different kinds of images on a Web page, most of which fall into two main cate gories: raster (or bitmap) and vector. A ***raster image*** is made up of pixels, whereas a ***vector image*** is made up of lines and curves based on mathematical expressions. A photograph is a type of ras ter image and is most often in JPG format. Other raster file formats that work well on Web pages are PNG, GIF, and BMP. A vector image is an illustration, such as a line drawing. Developers often convert vector file formats from programs like Adobe Illustrator or CorelDRAW, which aren't supported by Web browers, into PNG or GIF for Web display. An important difference between the two types of files is that raster images lose quality (they become pixelated) as you enlarge them, but vector images maintain quality even when enlarged.

The primary way to add images to an HTML document is with the img element. Like the anchor tag, the img tag does nothing by itself and requires attributes and values that specify the image the Web browser should display.

For example, to insert an image named redball.jpg that's in a subfolder called images, type this element:

```
<img src="images/redball.jpg" alt="Red ball
graphic" />
```

The image will display as long as the images subfolder is accessible. Both the src attribute and the alt attribute are required to be fully valid. The value of the alt attribute (short for alternate text) displays when a user hovers the mouse pointer over the image; in this case, the phrase "Red ball graphic" would display. The W3C requires the alt attribute for accessibility by people with disabilities. People with limited vision may use a screen reader, which reads aloud the alternate text for each image. Search engines also use the alt attribute to identify types of images and what's in them, since search engines can't "see" pixels in images.

As another example, to insert an image named bluelogo.png that's accessible from another Web site, type the following element:

```
<img
src="http://www.example.com/mrkt/images/bluel
ogo.png"  alt="Company XYZ blue logo" />
```

The img element uses several attributes, which are described in Table 2-3.

img element attributes

ATTRIBUTE	VALUE	D
src	URL	Specifies the image's location, such as a path or URL
alt	Text	Specifies alternate text for the image, which displays when the user hovers the mouse pointer or other pointing device over the image
height	pixels	Specifies the height of an image
width	pixels	Specifies the width of an image
ismap	ismap	Specifies an image as a server-side image map
usemap	#mapname	Specifies an image as a client-side image map (which is a picture with defined areas that are clickable links)

Using the **figure** and **figcaption** Elements

Two new graphics-related elements introduced in HTML5 are the figure and figcaption elements. The figure element specifies the type of figure you want to use in an HTML document, such as an illustration or photo. The figcaption element provides a caption for the figure.

The ***figure** element* specifies the type of figure you're adding, such as an image, diagram, photo, and so on. This element provides a major benefit: the ability to easily add multiple images side by side. With HTML 4, doing so requires a good bit of markup. The ***figcaption** element* is

optional. It adds a caption to an image on a Web page, and you can display the caption before or after the image.

The following markup uses the figure element, specifies the width and height of the image, and adds a caption. The result is shown in Figure 2-10:

```
<figure>
 <img src="doghappy.jpg" alt="Happy dog"
 width="100" height="125" />
          <figcaption>Happy dogs are good dogs</figcaption>
</figure>
```

Figure 2-10

Using the figure and figcaption elements to display an image with a caption

The following markup is for a figure with multiple images that share a single caption, the results of which are shown in Figure 2-11:

```
<figure>
 <img src="doghappy.jpg" alt="Happy dog"
 width="100" height="125" />
 <img src="dogpaws.jpg" alt="Happy dog"
 width="100" height="125" />
 <img src="dogwalk.jpg" alt="Happy dog"
 width="100" height="125" />
 <figcaption>Happy dogs are good dogs</figcaption>
<
```

Figure 2-11

U

i

Illustration: © MightyIsland/iStockphoto

s: © MightyIsland/iStockphoto

<figure>

Figure 2-12

The Web page with an image

MORE INFORMATION

To learn more about displaying images on Web pages, go to http://bit.ly/Kgg1ab. You can find out more about image maps at http://bit.ly/hincW5.

DISPLAY AN IMAGE IN A WEB PAGE

GET READY. To display an image in a Web page, perform the following steps: 1. Locate a JPG, PNG, GIF, or BMP file on your computer to use in this exercise. The image can depict anything you want, but something to do with pets would be most appropriate. 2. In your editing tool, open **L2-pet-mark.html**.

3. Remove the <mark> tags from the first paragraph.

4. Insert the following markup after the h1 element, leaving a blank line before and after it, and replacing dogwalk.jpg with your own image file:

6. Leave the editing tool and Web browser open if you're continuing immediately to the next exercise.

USE THE FIGURE AND FIGCAPTION ELEMENTS

GET READY. To display an image in a Web page, perform the following steps: 1. Locate two additional JPG, PNG, GIF, or BMP files to use in this exercise. The image can depict anything you want, but something to do with pets would be most appro priate. You should have three images to work through these steps.

2. In your editing tool, open **L2-pet-image.html** if it's not already open.

3. Replace the markup for the figure that follows the h1 element with the following, replacing the image file names (doghappy.jpg, dogpaws.jpg, and dogwalk.jpg) with your image file names:

```
<figure>
 <img src="doghappy.jpg" alt="Happy dog"
 width="100" height="125" />
 <img src="dogpaws.jpg" alt="Dog paws"
 width="100" height="125" />
                 <img src="dogwalk.jpg" alt="Walking a dog"
 width="100" height="125" />
              <figcaption>Happy dogs are good dogs</figcaption>
</figure>
```

4. Save the file as **L2-pet-multpimage.html** and view it in a Web browser. The page should look similar to Figure 2-13.

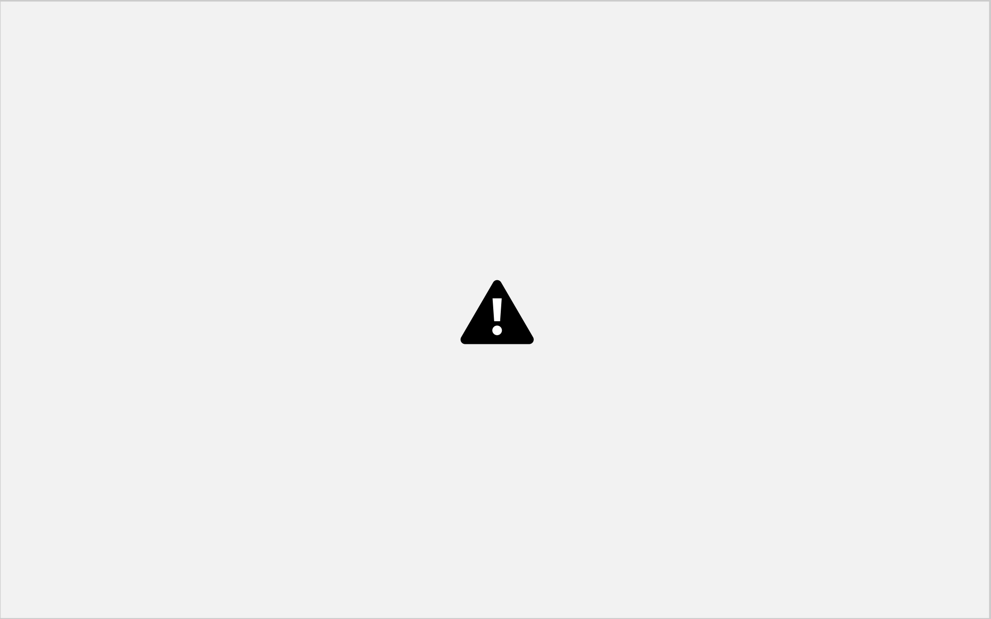


Figure 2-13

The Web page with multiple images and a caption noitartsulli

5. Close the **L2-pet-multpimage.html** file. Leave the editing tool and Web browser open if you're continuing immediately to the next exercise.

Creating Graphics with Canvas

The ***canvas element*** is new in HTML5 and creates a container for graphics, and uses JavaScript to draw the graphics dynamically.

With canvas, the Web page becomes a drawing pad, and you use JavaScript commands to draw pixel-based shapes on a canvas that include color, gradients, and pattern fills. Canvas also enables you to render text with various embellishments, and animate objects by making them move, change scale, and so on.

*

Developers use canvas to create online games, rotating photo galleries, stock tickers, and much more. The canvas element graphics and animation functions are intended to provide quality similar to those in Flash movies.

CANVAS BASICS

To use canvas, you first define a canvas in HTML. The basic syntax for the canvas element is as follows:

```
<canvas id="smlRectangle" height="100"
width="200"></canvas>
```

This element creates your drawing pad. The canvas element requires the id attribute to reference the canvas in JavaScript and to add it to the Document Object Model (DOM). You should also specify the dimensions of the canvas—the height and width—which are in pixels. JavaScript works with the two-dimensional (2D) application programming interface (API) to actually draw items on the canvas.

USE THE CANVAS TO CREATE A SHAPE

GET READY. To use the canvas element to create a shape, perform the following steps: **1.** In your editing tool, type the following markup:

```
<!doctype html>

<html>
 <head>
 <meta charset="UTF-8">
 <title>Canvas Test</title>

<script>
 function f1() {
 var canvas =
 document.getElementById("smlRectangle");
 context = canvas.getContext("2d");
 context.fillStyle = "rgb(0,0,255)";
 context.fillRect(10, 20, 200, 100);
 }
 </script>
 </head>

<body onload = "f1();">

<canvas id="smlRectangle" height="100" width="200 ">
</canvas>

</body>
```

```
</html>
```

You can include JavaScripts inside the head element of your HTML document, or in an external file.

The onload attribute causes the JavaScript function in the script to run. This script first finds the element with the id smlRectangle:

```
var canvas = document.getElementById("smlRectangle");
```

You'll learn how to use JavaScript in Lessons 8, 9, and 10.

Figure 2-14

The Web page with a canvas shape

The context.fillStyle method fills the rectangle with a blue color using the RGB values 0, 0, 255. The context.fillRect method creates a 200-pixel wide x 100-pixel tall rectangle, positioned 10 pixels down and 20 pixels over from the upper-left corner of the canvas and fills it using the color specified by fillStyle.

2. Save the file as **L2-canvas.html** and view it in a browser. The shape should appear as shown in Figure 2-14.

3. If a blue rectangle doesn't appear, go to the W3C Markup Validation Service Web page at **http://validator.w3.org**. Upload **L2-canvas.html** and click **Check** to have the service check it. Fix any errors reported by the checker. Save the file again and view it in a browser.

4. Leave the file, editing tool, and Web browser open if you're continuing immediately to the next exercise.

CREATING AN OUTLINE OF A SHAPE

To create an outline of a rectangle without a fill color, use the **context.strokeRect** method. It uses the same values as **context.fillRect**. To modify the color of the outline (the stroke color), use **context.strokeStyle**. For example, to create a 200 x 100 pixel rectangular outline in red, use these methods in your JavaScript:

```
context.strokeStyle = "red";

context.strokeRect(10,20,200,100);
```

USE CANVAS TO CREATE THE OUTLINE OF A SHAPE

GET READY. To use the canvas element to create the outline of a shape, perform the following steps:

1. In your editing tool, save **L2-canvas.html** as **L2-canvas-stroke.html**.

2. Replace the fillStyle and fillRect code lines with the following:

```
context.strokeStyle = "red";

context.strokeRect(10,20,200,100);
```

3. Delete the width and height attributes from the canvas element in the body (after id="smlRectangle").

4. Save the file and view it in a Web browser. The shape should appear as shown in Figure 2-15.

Figure 2-15

The Web page with a canvas shape outline

5. Close the file but leave the editing tool and Web browser open if you're continuing immediately to the next exercise.

PROVIDING AN ALTERNATE IMAGE OR TEXT FOR OLDER BROWSERS

Some older browsers cannot render canvas drawings or animation. Therefore, you should add an image, text, or some other HTML content within the canvas element that will display if the drawing cannot. The "backup" content, also referred to as fallback content, won't display if canvas is supported. This example displays an image (smlRectangle.jpg) similar to that which a filled rectangle canvas would create:

```
<canvas id="smlRectangle" height="100" width="200">
 <img
src="http://www.example.com/images/smlRectangle.jpg"
alt="A blue rectangle" />
</canvas>
```

To display text instead of an image, you would insert text in place of the `<img>` tag.

ADD A FALLBACK TO YOUR HTML DOCUMENT

GET READY. To add a fallback to your HTML document, perform the following steps:

1. In your editing tool, open **L2-canvas.html** and save it as **L2-canvas-canvas-fallback.html**.
2. Replace the canvas element with the following:

```
<canvas id="smlRectangle" height="100" width="200">
 Your browser does not support the canvas tag.
</canvas>
```

3. Save the file and view it in the Internet Explorer 9 Web browser. You should see the canvas drawing.
4. Press **F12** to enter browser mode, click **Document Mode** on the menu bar, and select **Internet Explorer 7 standards**.

5. Press **F12** again. An error message appears, stating that it doesn't recognize a property or method. Close the error message. The browser window displays the fallback text, as shown in Figure 2-16.

Figure 2-16

Text displays if the browser doesn't support canvas

6. Leave the file, editing tool, and Web browser open if you're continuing immediately to the next exercise.

MORE INFORMATION

For more information on the canvas element, visit the Microsoft HTML5 Graphics Web page at http://bit.ly/M8ZNkf. The HTMLCenter Web site at http://www.htmlcenter.com/blog/rgb-color-chart/ lists RGB color codes.

Creating Graphics with SVG

Scalable Vector Graphics (SVG) is a language for describing 2D graphics in Extensible Markup Language (XML). XML is a cousin to HTML, and has played an important part of HTML 4.01 Web pages. SVG technology is not new, but HTML5 now enables SVG objects to be embedded in Web pages without using the `<object>` or `<embed>` tags. (All types of SVG graphics are referred to as objects, and SVG loads into the DOM.)

2.2 The main purpose of SVG, as its name implies, is to create scalable vector graphic shapes, but you can create images and text as well. Much like canvas, you can apply solid colors, gradients, and pattern fills to SVG objects, and copy and clone objects. You can also use SVG anywhere you would insert a PNG, JPG, or GIF. With SVG, you provide drawing instructions rather than an image file.

One of the major benefits of SVG is its flexibility. Its vector graphic changes size to fit the screen on which it's displayed, whether the screen is on a 32-inch computer monitor or a mobile device like a smartphone. Because only the XML that describes the SVG graphic is transmitted, even large images don't require a lot of bandwidth. This makes SVG handy for use as a Web page background without having to use the repeat property. (Most solid Web page backgrounds are actually a thin line that's repeated using a CSS style.) In addition, SVG can be indexed by search engines because it's created by XML.

You can include attributes such as color, rotation, stroke color and size, and so on, to each SVG object. The following markup can be included in an HTML file to create a purple ball:

```
<svg id="svgpurpball" height="200"
xmlns="http://www.w3.org/2000/svg">

    <circle id="purpball" cx="40" cy="40"
r="40" fill="purple" />

</svg>
```

The cs, cy, and r attributes help to define the circle by defining the center x and y points and radius. SVG has a plethora of attributes, which help you create all kinds of shapes. The attributes are available on the W3C Web site at http://www.w3.org/TR/SVG/attindex.html.

MORE INFORMATION

For more information on SVG, visit the Microsoft SVG Web page at http://msdn.microsoft.com/en-us/library/gg589525(v=vs.85).aspx.

CREATE AN SVG VECTOR GRAPHIC

GET READY. To create a simple SVG vector graphic, perform the following steps:

1. In your editing tool, type the following markup:

```
<!doctype html>
<html>
 <head>
 <meta charset="UTF-8">
 <title>SVG Star</title>
 </head>
<body>
          <svg xmlns="http://www.w3.org/2000/svg" version="1.1">
             <polygon points="100,10 40,180 190,60 10,60 160,180"
               style="fill:aqua;stroke:orange;stroke-width:5;
fill-rule:evenodd;"/>
</svg>
</body>
</html>
```

The points attribute defines the x and y coordinates for each corner, or "point," of the polygon. The fill-rule determines how the inside of the polygon is filled.

2. Save the file as **L2-SVG.html** and view it in a Web browser. The page should look similar to Figure 2-17. If the page doesn't appear, check it using the W3C Markup Validation Service at **http://validator.w3.org** and fix any errors.

Figure 2-17

The Web page with an SVG shape

3. Change a few of the polygon point parameter values. Save the file as **L2-SVG-test.html** and view it in a Web browser. For example, changing the first parameter value from 100 to **50** produces the polygon shown in Figure 2-18.

Figure 2-18

Changing even one parameter value changes the shape of the object

4. Delete **fill-rule:evenodd;**, save the file, and then view it in a Web browser. Compare the polygon to Figure 2-17.

5. Close any open data files. Leave the editing tool and Web browser open if you're continuing immediately to the next exercise.

When to Use Canvas Instead of SVG

2.2

There are no hard and fast rules for choosing to use canvas or SVG. Your choice depends mainly on the nature of your project, and your skill level in one or the other.

The following are some considerations that will help you make the right decision: • If the drawing is relatively small, use canvas.

• If the drawing requires a large number of objects, use canvas. SVG begins to degrade as it continually adds objects to the DOM.

• Generally, use canvas for small screens, such as those on mobile devices. As the size of the screen increases and more pixels are needed, canvas begins to pixelate so use SVG. • If you must create highly detailed vector documents that must scale well, go with SVG. • If you are displaying real-time data output, such as maps, map overlays, weather data, and so on, use canvas.

A tip from Microsoft: Think of canvas as being similar to Microsoft Paint. You can draw images using shapes and other tools, and the result is pixel based. Think of SVG as being sim ilar to an Office PowerPoint slide, which uses scalable objects.

MORE INFORMATION

For more information on how to choose the best drawing method—canvas or SVG—go to http://msdn.microsoft.com/en-us/library/ie/gg193983(v=vs.85).aspx.

■

THE BOTTOM LINE

HTML5 introduces the audio and video elements, which do away with the need for plug-ins or media players to listen to music or watch videos via a Web browser.

The audio element and video element are two of the major changes in HTML5, enabling you to provide multimedia from a Web browser without the need for plug

2.3

2.3

ins, such as those for Microsoft Windows Media Player, Microsoft Silverlight, Adobe Flash, or Apple QuickTime. That means users can simply open an HTML5-supported browser to listen to music or audio books, enjoy rich sound effects, and watch video clips or movies.

The HTML5 specification includes the `<video>` and `<audio>` tags to incorporate multimedia. The following sections cover each in detail.

Understanding and Using Video Tags

You use the `video` element along with the `src` attribute to designate a video file to be played in an HTMLt document. Including he `height` and `width` attributes enables you to control the size of the window in which the video displays.

The ***video** element* enables you to incorporate videos in HTML documents using minimal code. The structure for embedding video is simple. The following is an example of the markup for adding an MP4 file to a Web page:

```
<video src="intro.mp4" width="400" height="300">
</video>
```

The `src` attribute points to the name of the video file (in this case, video.mp4) to be played. The height and width attributes specify the size of window in which the video will display.

Other attributes are available that you can add for control of the video:

- **poster:** Displays a static image file before the video loads
- **autoplay:** Start playing the video automatically upon page load
- **controls:** Displays a set of controls for playing, pausing, and stopping the video, and controlling the volume
- **loop:** Repeats the video

Using all of the controls listed above, the markup would look similar to this:

```
<video src="/videos/intro.mp4"
 width="400" height="300
 poster="78704-splash.jpg"
 autoplay="autoplay"
 controls="controls"
 loop="loop">
</video>
```

Notice that this markup refers to an MP4 video file. Other popular Web video formats also include H.264, OGG, and WebM, although WebM is used less than 10 percent of the time. Along with a video format, you should also specify the *codec*, which is a technology used for

compressing data. ***Compression*** reduces the amount of space needed to store a file, and it reduces the bandwidth needed to transmit the file. ***Video compression*** reduces the size of video images while retaining the highest quality video with the minimum bit rate. All of this makes for better performance.

In a nutshell, the main video formats along with codecs (for the last two) are:

- MP4 or H.264
- OGG + Theora with Vorbis audio
- WebM + VP8

A best practice is to use the type attribute to specify the video format. You should also use the codecs attribute to specify the codec(s), if applicable. Sample markup is shown as follows:

```
<video
 width="400" height="300"
 poster="78704-splash.jpg"
 autoplay="autoplay"
```

```
controls="controls"
loop="loop">

          <source src="intro.mp4" type="video/mp4" />

</video>
```

The `<source>` tag is being used as content of the video element so that the type attribute can be set and so that the multiple format option is available.

Not all video formats are supported by all browsers, although MP4/H.264 is the most widely used by both Web browsers and mobile devices. (The HTML5 Video Web page at http://www.w3schools.com/html5/html5_video.asp displays a table showing which video formats work for what browser. The table is updated regularly.) To help make your video viewable by the majority of browsers and devices, you can use the source attribute to include multiple formats in your markup. This example shows the same video available in two formats, and the OGG format specifies codecs:

```
<video
     width="400" height="300" poster="image.png"
autoplay="autoplay"
controls="controls"
loop="loop">

<source src="video.mp4" type="video/mp4">

<source src="video.ogg" type='video/ogg;
codecs="theora, vorbis"'>

</video>
```

WORK WITH THE VIDEO ELEMENT

GET READY. To work with the HTML5 video element, perform the following steps:

1. Locate a video clip, and an image file to use as a poster. If you don’t have a video clip, search for a public domain MP4 file on the Web and download it. Save the video and image files to your HTML5 folder.
2. In your editing tool, create an HTML file with the following markup. Substitute appropriate file names for your image file and video clip. Change the type attribute, if necessary, and replace sample.mp4 with the name of your video file.

```
<!doctype html>
<html>
 <head>
 <meta charset="UTF-8">
 <title>Video Test</title>
 </head>
<body>
 <video
 width="400" height="300"
 poster="sample.jpg"
 autoplay="autoplay"
 controls="controls">
 <source src="sample.mp4" type="video/mp4" />
 </video>
</body>
</html>
```

3. Save your file as **L2-video.html**.

4. Go to the W3C Markup Validation Service Web page at **http://validator.w3.org**. Upload **L2-video.html** and click **Check** to have the service check it. Fix any errors reported by the checker that relate to missing tags or typos, if any.

5. Open the HTML file in a Web browser. Does the video play automatically? Do the controls appear? You should open the L2-video.html file in a variety of Web browsers as a test.

6. In your editing tool, delete the **autoplay** line and replace controls="controls" with simply **controls**.

7. Save the file again and validate it. It should validate with no errors. That indicates that HTML5 allows you to use a shorthand method of specifying the controls attribute. The same principle applies to the autoplay and loop attributes.

8. Leave the editing tool and Web browser open if you're continuing immediately to the next exercise.

Understanding and Using Audio Tags

The HTML5 audio element works much like the video element but for sound only. To use the audio element, include the <audio> tag and a path to the file on your hard drive or a uniform resource locator (URL) that points to the audio file.

2.3

*

The ***audio*** *element* enables you to incorporate audio, such as music and other sounds, in HTML documents. You can include the same control-related attributes as the video element: autoplay, controls, and loop. the following example shows just the controls attribute included:

```
<audio src="sample.mp3" controls="controls">
```

```
</audio>
```

If you use the autoplay attribute so that the audio plays automatically when the Web page loads, use a short clip such as a sound effect. Many Web visitors dislike automatic sound and prefer to have more control.

The three primary types of audio files supported by popular browsers are OGG, MP3, and WAV. However, not every browser supports every audio file format, at least not today. For the most part, MP3 is the best choice for multiple browser compatibility.

To help ensure your audio plays on the majority of browsers and devices, use the source attribute to include multiple formats in your markup. This example shows the same audio file available in two formats:

```
<audio controls="controls">
        <source src="sample.ogg" type="audio/ogg" />
       <source src="sample.mp3" type="audio/mp3" />
</audio>
```

You can find a lot of free audio files, which are also royalty and copyright free, at http://flashkit.com. This is a good resource for learners, and for developers who may need a sound effect for a project. Another source is the Public Domain Sherpa Web site at http://www.publicdomainsherpa.com/public-domain-recordings.html. You can also make your own recordings using your computer and recording software. Windows 7 includes the Sound Recorder, which lets you save audio files in WAV format.

WORK WITH THE AUDIO ELEMENT

GET READY. To work with the HTML5 audio element, perform the following steps:

1. Locate an audio clip.
2. In your editing tool, create an HTML file with the following markup. Substitute the appropriate file name for your audio clip.

```
<!doctype html>
<html>
 <head>
 <meta charset="UTF-8">
 <title>Audio Test</title>
 </head>
<body>
        <audio src="sample.mp3" controls="controls">
 </audio>
</body>
</html>
```

3. Save your file as **L2-audio.html** and view it in a browser. You should see something similar to Figure 2-19. Because we didn't include the autoplay attribute in this example, you need to click the Play button to start the audio clip.

Figure 2-19

HTML5 default audio controls
in a Web browser

4. If the audio controls don’t appear, go to the W3C Markup Validation Service Web page at **http://validator.w3.org**. Upload **L2-audio.html** and click **Check** to have the service check it. Fix any errors reported by the checker that relate to missing tags or typos, if any.

5. Save the file again and open it in a Web browser. Play the audio clip.

6. Close any open files, including the editing tool and Web browser.

MORE INFORMATION

For more information on incorporating multimedia into HTML5 Web pages, and the audio and video elements in particular, go to http://msdn.microsoft.com/en-us/library/ie/hh771805(v=vs.85).aspx.

SKILL SUMMARY

- Hypertext Markup Language (HTML) uses markup to describe content for display on a Web page.
- An element is the combination of tags and the content they enclose. You may need to use special characters on a Web page, which requires character encoding.
- Every Web page requires the doctype declaration at the top of the page.
- HTML5 uses most of the same elements and attributes specified in HTML 4, and has introduced some new tags, modified the preferred usage of others, and no longer supports certain elements. New text-related elements include command, mark, time, meter, and progress. A few of the deprecated elements are basefont, center, font, and strike.
- Use the img element to display linked images in a Web page. The images can be located with the Web pages HTML files, usually in an images subfolder, or on a different server or Web site.
- The figure and figure caption elements are new to HTML5 and give you more control or the type of image you are displaying and the ability to include captions.
- The canvas element is used for drawing, rendering, and manipulating images and graphics dynamically in HTML5.
- Scalable Vector Graphics (SVG) enables you to create scalable objects that resize to best fit the screen on which they’re viewed, whether a PC screen or a smartphone.
- HTML5 introduces the audio and video elements, which do away with the need for plug-ins or media players to listen to music or watch videos via a Web browser.

Fill in the Blank

Complete the following sentences by writing the correct word or words in the blanks provided.

1. An HTML tag that doesn’t require an end tag is called a(n) tag.

2. A(n) works with an element to describe data in enough detail for rendering.

3. The is a declaration that is found at the very top of almost every Web page.

4. A element or attribute has been removed from the list of available HTML elements according to the W3C.

5. A image is made up of pixels, whereas a image is made up of lines and curves based on mathematical expressions.

6. New to HTML5, the element specifies the type of figure you're add ing, such as an image, diagram, photo, and so on.

7. The element adds a caption to an image on a Web page, and you can display the caption before or after the image.

8. Using the element, the Web page becomes a drawing pad, and you use JavaScript commands to draw pixel-based shapes on a canvas that include color, gradi ents, and pattern fills.

9. is a language for describing 2D graphics in Extensible Markup Language (XML).

10. The HTML5 element and elements enable you to provide multimedia from a Web browser without the need for plug-ins.

Multiple Choice

Circle the letter that corresponds to the best answer.

1. Which of the following tags are required on every Web page? (Choose all that apply.)

a. <html>
b. <head>
c. <title>
d. <body>

2. Which of the following is the syntax for creating a hyperlink in HTML?

a. <link href="http://www.example.com">link</a>
b. <a href="http://www.example.com"> link text</a>
c. <link>http://www.example.com</link >
d. <http://www.example.com>

3. Which HTML5 element defines a command button that users click to invoke a command?

a. <objectbut>
b. <combutton>
c. <command>
d. <cbutton>

4. Which HTML5 element enables you to highlight blocks of text in an HTML document?

a. <mark>
b. <highlight>
c. <emphasis>
d. <yellow>

5. Which of the following tags are deprecated in HTML5? (Choose all that apply.)

a. <big>
b. <center>
c. <font>
d. <time>

6. Which tag is used with the <figure> tag to display an image?

a. <img>
b. <src>
c. <fig>

d. <a>

7. Both canvas and SVG require which of the following?

a. Microsoft Silverlight
b. An external drawing program, such as Microsoft Paint
c. A large amount of storage space or bandwidth
d. JavaScript

8. When deciding whether to use canvas or SVG, which of the following considerations are true?

a. If the drawing is relatively small, use SVG.
b. Generally, use canvas for small screens and SVG for larger screens.
c. If the drawing requires a large number of objects, use SVG.
d. If you must create highly detailed vector documents that must scale well, go with canvas.

9. Which of the following is the general format of the video element?

a. <movie src="file.mp4" width="*X*" height="*Y*">
b. <movie href="file.mp4" width="*X*" height="*Y*">
c. <video src="file.mp4" width="*X*" height="*Y*">
d. <video href="file.mp4" width="*X*" height="*Y*">

10. Which of the following is the general format of the audio element?

a. <audio src="sample.mp3" controls="controls">
b. <audio href="sample.mp3" controls>
c. <sound src="sample.mp3" controls>
d. <sound href="sample.mp3" controls="controls">

True / False

Circle T if the statement is true or F if the statement is false.

T F 1. The canvas element requires JavaScript to create shapes.
T F 2. Creating an SVG object in HTML5 does not require JavaScript.
T F 3. The audio element can provide playback controls with a single attribute.
T F 4. Deprecated elements cannot render in an HTML5-supported browser.
T F 5. The most popular format for audio files is MP4.

Scenario 2-1: Correcting Simple Markup Errors

Geraldine, the assistant to the company owner, is learning HTML. Her markup as shown below isn't rendering as she expected. The boldface doesn't stop after "Thursday." The image of the company logo doesn't display, even though it's saved in her images subfolder like all of her other images. The alternate text doesn't display either when she hovers her mouse pointer over the image placeholder. What do you tell her?

```
<!doctype html>
<html>
 <head>
 <meta charset="UTF-8">
 <title>Internal</title>
```

```
                    </head>

<body>
<h1>Staff Meeting</h1>
<img src="cologo.jpg" olt="Company logo" />
        <p>Report to the <strong>Blue Conference Room</strong> at
        <strong>10:00 a.m.</strong> on <strong>Thursday<strong> for an
emergency staff meeting.</p>
</body>
</html>
```

Scenario 2-2: Working with Symbols

Petra is formatting some accounting-related documents to be hosted on the company's intranet. She says the dollar signs and percent symbols look fine when she views them in one browser, but only garbage characters appear when she views the documents on a different browser. What should she do?

■

Scenario 2-3: Canvas or SVG?

M.A. is a graphic artist at ClickTick Watches, an upscale wristwatch manufacturer. She has been asked to refresh the company logo and create it using a tool that scales well whether the image is viewed on laptops or smartphones. She has also been tasked with creating interactive graphs for sales staff to use on their slate or tablet devices. She wants to keep her skillset cur rent by learning as much as possible about HTML5 technologies, but doesn't know whether to focus on canvas or SVG for these projects. What do you suggest?

Scenario 2-4: Selecting Appropriate Web Video Formats and Codecs

Sammy is responsible for setting up meetings for employees of Clear Blue Resorts. He wants to post a video from the CEO, who is overseas reviewing possible locations for new resorts, to the intranet for the upcoming employee appreciate party. He knows Clear Blue standardized on Internet Explorer 9, and he has heard that he can easily display video in HTML5 but doesn't know where to start. What do you tell Sammy?

Building the User LESSON

Interface by Using HTML5: Organization,

Input, and Validation

EXAM OBJECTIVE MATRIX

S /CONCEPTS MTA EXAM OBJECTIVE MTA EXAM OBJECTIVE N

Choosing and Configuring Choose and configure 2.4
HTML5 Tags to Organize HTML5 tags to organize
Content and Forms content and forms.

Choosing and Configuring HTML5 Choose and configure HTML5 2.5
Tags for Input and Validation tags for input and validation.

article element **aside** element
autofocus attribute automatic validation client-side validation **datalist** element **email** attribute
footer element
form input
global attribute **header** element **menu** element
nav element
ordered list
pattern attribute
placeholder text
required attribute
section element
semantic markup
server-side validation
table
unordered list
validation
Web form

One of your new tasks as an intern at Malted Milk Media is to create a Web form that restricts what a user can enter into the form fields and validates the input. To prepare to create the Web form, you must first learn how best to organize or structure the markup using new HTML5 elements.

THE BOTTOM LINE

HTML5 introduces several new elements for organizing content and forms. They represent the new semantic markup that's an important part of HTML5.

HTML5 markup introduces many new markup tags for organizing the structure of HTML documents, which makes documents easier to create and modify. The new tags have more intuitive names than similar constructs in previous HTML specifications; the tags are named more appropriately for the part of the page they apply to, such as <header>, <section>, and <footer>.

HTML5 has also streamlined table creation, moving many of the table attributes that affect width, cell padding, and vertical and horizontal alignment to the CSS file.

Understanding Semantic HTML

Semantic markup uses tag names that are intuitive, making it easier to build and modify HTML documents, and for Web browsers and other programs to interpret.

2.4

*

One of the very handy new features of HTML5 is the use of ***semantic markup***, which gives better meaning, or definition, to several tags so they make more sense to humans, programs, and Web browsers. As mentioned in Lesson 2, not all HTML tags have been replaced or updated for HTML5, but some new tags introduced in HTML5 make the work of creating Web pages a lot easier.

In HTML 4.01 and prior specifications, a developer creating the structure of an HTML document uses the <div> tag frequently throughout. The <div> tag often includes a class or ID attribute, which may also include CSS styles such as background-color, height, and width. A simple example of a <div> tag is:

```
<div id="header" > This is a header </div>
```

class and id are ***global attributes***, which means they can be used with any HTML element. You can see the complete list of global HTML attributes at http://dev.w3.org/html5/markup/global-attributes.html.

The div element alone doesn't have much meaning without the id or class attribute. Even the ID can be assigned a value of your choice, such as "header", "header_inner", "slo gan", "content", "style", and many more. An example from an HTML 4.01 document is shown as follows:

```
<div id="header">
 <div id="header_inner">
 <img src="images/doghappy.jpg"
 alt="Attaboy Pet Services" />
 <div id="slogan">Happy dogs are good dogs</div>
 </div>
</div>
```

HTML5 uses simpler tags to replace many of the div tags, some of which are shown in Figure 3-1.

HTML 4.01 HTML 5

Figure 3-1

A comparison of document parts marked up in HTML 4.01 and HTML5

2.4

Table 3-1

New HTML5 markup tags for organizing content and creat ing structure

Notice how HTML5's semantic markup gives more specific meaning to parts of an HTML document, making the structure easier to understand.

Using Tags to Add Structure to an HTML Document

*

New HTML5 elements for structuring and organizing content in an HTML document include header, footer, section, nav, article, and aside.

Now that you understand semantic markup, let’s look at several of the new HTML5 elements for organizing documents. Table 3-1 lists new HTML5 structure-related tags and describes them.

TAG D

<address> Defines an area for contact information for a page or section

<article> Defines an article, such as a magazine or newspaper article, blog post, or similar content

<aside> Defines content that’s separate from but related to the page content; simi lar to a sidebar in book chapters and magazine articles

<details> Contains additional details pertinent to text around it; creates an interac tive widget a user can display or hide

<footer> Defines a footer for a document or section; may include the document author, contact information, copyright information, and links to terms of use

<header> Defines a header for a document or section; may contain introductory con tent or navigation links

<hgroup> Groups headings and subheadings (using the <h1> to <h6> tags) for multi-level headings

<nav> Defines a block of navigation links

<section> Defines a section in a document, such as chapters, parts of a thesis, or parts of a Web page whose content is distinct from each other

<summary> Defines a visible heading for a details element; user can click to display or hide information

<wbr> Defines a possible line break; when a word is very long, or you’re con cerned the browser will break a line at the wrong place, you can use the <wbr> element to break the word or line appropriately

Figure 3-2

Multiple instances of the header element

As you learned in Lesson 1, the HTML5 standard won’t be finalized for several years, which means changes to the specification are still occurring. The major Web browsers, like Microsoft Internet Explorer and Mozilla Firefox, support many HTML5 elements but not all of them. The “When Can I Use” Web site at http://caniuse.com/ is an excellent source for determining which browsers support specific HTML5 features. The Web site is updat ed regularly, so you should make it a part of your essential HTML5 resources. In addition, you can test any browser’s support for HTML5 by browsing to the HTML5 Test Web site at http://html5test.com.

Let's look more closely at the HTML5 tags shown in Figure 3-1, which are the most com monly used structure-related HTML5 tags.

THE HEADER AND FOOTER ELEMENTS

The ***header* element** defines a header for a document, section, or article. In HTML 4.01, you use the header div as mentioned in the previous section (`<div id="header">`). The ***footer* element** defines a footer for a document or section, and typically contains informa tion about the document or section, such as the author name, copyright data, links to related documents, and so on. The `footer` element doesn't automatically appear at the bottom (or foot) of the document—you need to use CSS to instruct the browser where to display the footer. Footers that appear at the bottom of every Web page or document are known as "sticky footers."

An example of an article with a `header` tag and a `footer` tag is as follows:

```
<article>
 <header>
 <h1>Learning HTML5</h1>
 <h2>The New Elements</h2>
 </header>
 <p>New HTML5 tags make Web page and application
 development easier than ever.</p>
 <footer>
 <p>Published: <time datetime="2012-09-
 03"September 3, 2012</time></p>
 </footer>
</article>
```

2.4

Table 3-2

Situations in which you should not use the section element

Like the `div` element, you can use the header and footer elements multiple times in an HTML document, as shown in Figure 3-2.

HTML 5

*

The new HTML5 struc ture-related tags don't replace the <div> tag entirely, but HTML5 tags greatly reduce the number of `<div>` tags needed in an HTML document.

THE SECTION ELEMENT

The ***section* element** defines a section in a document, such as a chapter, parts of a thesis, or parts of a Web page whose content is distinct from each other. The WC3 specifies uses for the section element to differentiate it from other structure-related elements, mainly that it contain at least one heading and that it define something that would appear in the docu ment's outline. For example, you should use the `section` element to divide different parts of a one-page Web site or to create a portfolio of images. The following is an example of a simple section:

```
<section>
 <h1>Eight Count</h1>
 <p>Hip-hop dance instructors often teach moves
 that have eight counts per set.</p>
</section>
```

Table 3-2 lists situations in which you should avoid using the **section** element, and provides the better technique.

SITUATION U

Separate content that is independent from the rest of the content article on the Web page or document

Plan to syndicate a block of content article Create a

sidebar aside

Wrap and position multiple sections that are not related to div
each other

Add a drop shadow to or border around an item div

Knowing when to use the **<section>** tag versus a different element can be tricky at times. When you're working on an HTML document and are unsure which element to use, browse the W3C HTML5 specification or research the Web to see how other developers have handled a similar situation.

When defining a section header, which may contain h1 through h6 headings, you can use the **hgroup** element to group headings. The **hgroup** element affects organization but not presentation. Consider using **hgroup** when you have a heading and a subheading one right after the other, as follows.

```
<section>
 <hgroup>
 <h1>Hip-Hop Dance Routines</h1>
 <h3>The Eight-Count Method</h3>
 </hgroup>
 <article>
 <p>Hip-hop dance instructors often teach
 moves that have eight counts per set.</p>
 </article>
</section>
```

This markup would appear in a Web page as shown in Figure 3-3.

Figure 3-3

Using hgroup to group headings in an HTML document

*

Remember, you have several choices of tools to use for creating
HTML documents. For the PC, consider the Notepad or Notepad++ text editors, the HTML Kit or KompoZer HTML editors, or development tools like Microsoft Visual Studio, Visual Studio for Web, or Microsoft Expression Web.

CREATE AN HTML DOCUMENT WITH A HEADER, SECTIONS, AND A FOOTER

GET READY. To create an HTML document using the HTML5 header, section, and footer elements, perform the following steps:

1. Using an HTML editor or app development tool and a Web browser, create a simple HTML document that incorporates the <header>, <section>, and <footer> tags.

Include two sections, and be sure to include at least one h1 element within the sec tions. You can include images if you want. The markup might look like the following:

```
<!doctype html>
<html>
<head>
 <meta charset="utf-8" />
 <title>My Page</title>
</head>
<body>
<header>
 <h1>Selecting a Concert Style</h1>
</header>
<section>
 <h1>Symphonies</h1>
 <p>A symphony is a type of musical composition
generally   performed by a full orchestra.</p>
</section>
<section>
 <h1>Raves</h1>
 <p>A rave is a gathering of people who listen and dance
to  music, especially electronic music, usually performed
by a  live band or live DJs.</p>
</section>
<footer>
 <p>Author: Nathaniel Becker</p>
</footer>
</body>
</html>
```