Java Methods

A & AB

Object-Oriented Programming
and
Data Structures

Maria Litvin
Phillips Academy, Andover, Massachusetts

Gary Litvin
Skylight Software, Inc.

Skylight Publishing
Andover, Massachusetts

Skylight Publishing
9 Bartlet Street, Suite 70
Andover, MA 01810

web: http://www.skylit.com
e-mail: sales@skylit.com
support@skylit.com

Library of Congress Control Number: 2005910949

ISBN-10: 0-9727055-7-0
ISBN-13: 978-0-9727055-7-8

Appendix C: HTML Tutorial

C.1 Prologue

HTML, or HyperText Markup Language, is a simple tool for formatting documents, especially web pages. HTML is useful to Java programmers in three ways:

- Java applets run on web pages (see Section C.9)
- HTML tags, in particular `<code>`, can be embedded in Javadoc comments [1]
- HTML tags can be embedded in text in `JLabel`s, `JButton`s, and other *Swing* components [1, 2]

So it is important for us to get at least a basic idea of what HTML is all about.

The term *hypertext* refers to a text fragment or document in which certain words or phrases act as "hot links": when you "touch" such a link (click on it with a mouse or literally touch it on a touch screen) it takes you to a specified place — in the same text, in another document on your computer, or, with the Internet, to a document on a computer halfway around the world. This powerful method of viewing and browsing through information is possible only with computers. In 1987, Apple Computer, Inc. released a Macintosh software application called *HyperCard* that popularized the hypertext concept and gave non-programmers a tool for creating hypertext applications.

Markup implies that the text is formatted by means of embedded commands or "*tags*." Marked-up text is called *HTML source*. A program that can interpret HTML files looks at the tags and renders the document on the screen or in a printout according to the formatting commands in the document (and also according to the capabilities of a particular monitor or printer). For example, if you wanted a word or a phrase to appear in italics, you would mark that phrase with an opening `<i>` tag and a closing `</i>` tag in the HTML source, as in:

```
The term <i>hypertext</i> refers to
```

It will be interpreted and shown as

HTML documents can be created using a general-purpose text editor, a word processor, or a program editor. One approach is to first put all the textual information in, spell-check the text, then add HTML tags. You can also take an existing document, say in *MS Word* format, save it as "text only," then add HTML tags to it. Many programs can convert text formatted with a word processor into HTML. For example, *MS Word* has a "save as HTML" feature.

> **HTML files usually have the extension `.html` or `.htm`. If you are using a text editor or a word processor to write them, make sure you save documents as "text only" type but give the saved file name the `.html` or `.htm` extension.**

HTML files can be interpreted by different programs, but primarily by Internet browsers (FireFox, Microsoft's *Internet Explorer*, and other browsers [1]). As HTML evolves and new browser releases appear, the HTML standard is by and large controlled by the most popular browsers and also by the World Wide Web Consortium (W3C) [1], which coordinates development of new technologies for the web. Other programs, including some word processors, can interpret HTML, too. But if you use a simple text editor (for example, *Notepad* in *Windows*) to create HTML documents, then you have to use a web browser to see how the final document will look, going back and forth between the editor and the browser to make adjustments and see the results. Specialized web authoring tools can generate HTML documents automatically in a WYSIWYG ("What-You-See-Is-What-You-Get," pronounced "wee-zee-wig," just like it looks) manner. Then you can make minor adjustments by editing the HTML source.

HTML was conceived in late 1980's by Tim Berners-Lee, [1] a computer scientist at CERN, an international high-energy physics research center in Switzerland. Berners-Lee was looking for a universal format that would allow users to create documents without specialized software or word processors, keep the formatting overhead in documents small, and allow the documents to travel over a network as ASCII text. The formatting commands also had to be suitable for displaying the same document on a variety of computers with different display capabilities: from simple text terminals to fancy graphics devices. In the early 1990s HTML quickly became the standard for the emerging network of computers that supported *http* (HyperText Transfer Protocol) — the network now called the *World Wide Web*.

> **A *web page* is basically an HTML document. A *web site* is a collection of related web pages and supporting files, such as images, audio clips, MP3 files, PDF (Portable Document Format) files, plain-text files, and so on, usually located on the same *host* (computer connected to the network).**

HTML is not a programming language and designing web pages in HTML is not programming. Although HTML tags can be viewed as commands or instructions of a sort, HTML has no facility for implementing algorithms, a process considered to be the key characteristic of programming. (Algorithms are explained in Chapter 4.) Writing an HTML file is more like formatting text in a word processor. If you want your web pages to perform more sophisticated tasks, you have to program these tasks in a programming language to be used in conjunction with HTML.

One such language is JavaScript [1, 2], a somewhat limited programming language specifically designed for adding some action to web documents. Its primary use is for validating and processing data that come from online forms embedded in web pages. For example, when you type your address and credit card information into a form on a web page, JavaScript functions may be used to check that the required fields are filled with reasonable values and that the credit card number is valid. JavaScript shares some syntactic elements with Java, but the similarity ends there. JavaScript deals primarily with objects embedded in web documents (such as fields in a form) and lacks the capabilities required for writing general-purpose programs. We do not go into HTML forms or JavaScript in this tutorial.

Naturally, you can find many HTML tutorials and on the web [1, 2]. Many web sites also have reference guides for HTML tags [1, 2, 3, 4, 5]. You can also learn from any web page: if you choose the `View source` menu option in your browser, you will see the document as a plain-text document with embedded HTML tags. Go to any simple web page and try it.

In this tutorial you will learn the following HTML features:

- The main structural elements of an HTML document
- Text layout (headings, paragraphs)
- Text formatting (font attributes, colors)
- Hyperlinks
- Using lists
- Embedding images into HTML documents
- Using tables
- Embedding Java applets into HTML documents

C.2 Example

Figure C-1 shows an example of a simple HTML document, and Figure C-2 shows its appearance in an *Internet Explorer* window.

Note the following features:

1. Each HTML tag is enclosed in angular brackets, `<...>`. Many opening tags have a corresponding closing tag that starts with a slash (as in `<xyz>` - `</xyz>`). Some closing tags, especially for the new paragraph mark `<p>`, may be omitted. Some tags, like `<img>`, do not use a closing tag.

2. HTML tags are not case-sensitive. Some people use all caps to make the tags stand out more, others use lower case.

3. Unless overridden by special tags, HTML interpreters ignore line breaks and whitespace characters (spaces, tabs, etc.) in the original document, replacing each whitespace sequence with one space. HTML interpreters basically treat the whole document as one long line of text. Special tags such as `<p>` (new paragraph), `<br>` (line break), and tags marking lists and tables control how the lines of text appear on the screen.

3. The whole document is enclosed between `<html>` and `</html>` tags that identify the type of the document for the browser.

4. The rest of the document is split into two parts: the "head" (enclosed between `<head>` and `</head>`) and the "body" (enclosed between `<body>` and `</body>`).*

* More formal HTML specification calls the opening tag and the closing tag and everything in between *a tag*, and defines formally which other tags may occur inside a given tag. In this terminology, an HTML document consists of one `html` tag, which contains `head` and `body` tags. Thus this terminology stretches the common use of the word "tag" a bit.

```
<html>

<head>
<title>Winter Track</title>
<meta name="Keywords" content="Andover, Winter Track, Athletics">
</head>

<body>

<center>
<h2>Girls' Winter Track</h2>

<table width="95%" cellpadding="10">
<tr>
<td width="50%">
Lead by captains Lucy Greene and Sean Scott, Indoor Track
completed another winning season studded by many brilliant
individual performances.  Rain, snow, sleet, or storm,
the girls practiced since early December under the cover of
the Case Memorial Cage and consequently took the lead over
competitors early on.  Phillips Academy ran against local
schools Haverhill, Tewksbury, Andover High School, Central
Catholic, and Chelmsford, falling only to the Tewksbury team.

</td>
<td valign="center" width="50%">
<img src="wintertrack.jpg" alt="Winter track meet"
     width="233" height="235">
</td>
</tr>
</table>

<p><i>By L.G.</i>

<br><br>

<hr width="80%" size="3" color="#0000A0">
<a href="athletics.html">Back to Athletics</a>
<hr width="80%" size="3" color="#000080">
</center>

</body>
</html>
```

Figure C-1. JM\Appx-C\wintertrack.html

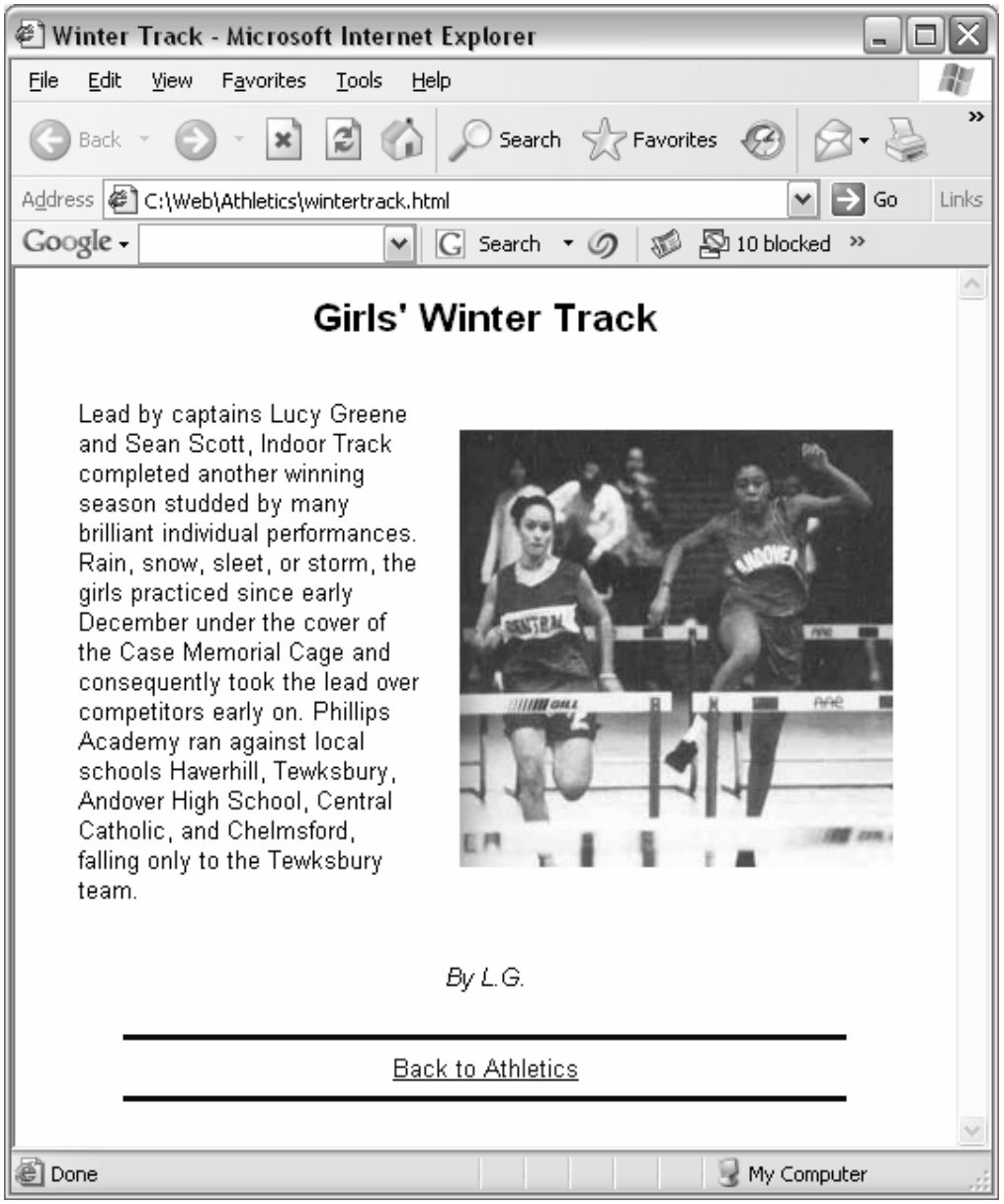


Figure C-2. The HTML document from Figure C-1 as it appears in *Internet Explorer*

5. Some tags may have attributes. For example, the <h2> tag that marks a header line (of the second rank) may have an attribute that centers the heading in the window:

```
<h2 align="center">Girls' Winter Track</h2>
```

   The <hr> ("horizontal rule") tag that makes a horizontal line may have optional attributes for the line width, "size" (thickness), and color:

```
<hr width="80%" size="3" color="#0000A0">
```

   In general, attributes have the form attribute="*value*". Some attributes (such as src in <img>) are required, while others (such as alt, width and height in <img>) are optional.

6. Certain characters, such as >, <, &, and ", cannot be used directly in HTML documents because they are reserved for HTML commands. They are represented by "escape sequences" such as >, <, &, and " in the HTML source. An escape sequence starts with an ampersand (&), ends with a semicolon, and has a standardized abbreviated name of a character in between. Escape sequences are also used for characters that are not on the keyboard, such as © for the copyright symbol, and special characters, such as   (non-breaking space). You can find a complete list of escape character sequences on the web [1].

C.3 Document Structure Tags

HTML document structure is shown in Figure C-3. The three main tags that help establish a document's structure are <html>, <head>, and <body>. As we mentioned above, the <html> tag is the outermost tag that identifies the document as an HTML document. The <head> ... </head> tag isolates information about the document itself rather than information to be displayed in the document. The head can contain <title>, one or several <meta> tags, and a few other tags. The <title> tag defines the document's title as it will be displayed in the browser window's title bar when you open the document. It is also used as the bookmark description when visitors bookmark your page in their browser, so it is important to choose a meaningful and descriptive title. For example, "Alphabetical Listing of HTML Tags" is a better title than "Tags."

An optional <meta> tag can list just about any information about your document. Its name attribute tells what information is included and its content attribute lists the information. You can write, for instance,

```
<meta name="my dog's name" content="Tessie">
```

but this may be not very relevant. A more conventional use is to enter the author's name and a few keywords to help web search engines find the page. For example,

```
<meta name="author" content="Mark Taggit">
<meta name="keywords" content="HTML, www, tutorial">
```

<meta> tags should not be viewed as just comments because they may be read by search engines. HTML has a special comment tag:

```
<!-- any text -->
```

that can be used to explain the HTML source. You can also temporarily disable pieces of HTML text and commands by surrounding them with a comment tag.

```
<html>

<head>
<title>title text</title>
<meta name="author" content="...">
<meta name="keywords" content="...">
</head>

<body attributes>
...
<address>
...
</address>
</body>

</html>
```

Figure C-3. HTML document structure

The information displayed in the document is placed between opening and closing <body> tags. The <body> tag may have some optional attributes that redefine the default text color, background color or background image, colors for active and followed links, and other attributes. For example,

```
<body text="coral" bgcolor="antiquewhite">
```

displays “coral” text on an “antiquewhite” background. You can find a complete list of HTML web-safe colors (with samples) on the web [1]; of course, all common color names are included. You can also specify color as a # sign followed by six hex digits — two hex digits (that is, a value from 0 to 255) for the intensity of each of color’s red, green, and blue components. For instance, `"#FFFFFF"` represents white and `"#000080"` represents dark blue.

It is customary to close the body of the document with an `<address>` tag that lists (in italics) the name and address of the organization or person responsible for the web page, the last revision or copyright date, and so on. For example:

```
<address>
Copyright &copy; 2006 by Nasty Loops, Inc.<br>
<br>
This entire site is protected by copyright.<br>
All rights reserved.  By looking at this site, you agree
to be subject to the &quot;Terms and Conditions Governing
Use and Access to Nasty Loops Online.&quot;
</address>
```

(The Internet culture is open and free, and Internet users normally expect to see friendly sites with no complicated licensing terms or user agreements. However, copying the entire web documents and installing them on your own web site is unethical and illegal.)

C.4 Text Layout and Formatting Tags

The HTML tags useful for text layout are summarized in Figure C-4.

`<h1>` through `<h6>` tags can be used to display document and section or subsection headings of different levels: from `<h1>` (the biggest font) to `<h6>` (the smallest). These headings do not have to be used in any particular order and, unlike in some word processors, do not automatically keep track of section or subsection numbers. The heading tags can include a horizontal alignment attribute. For example:

```
<h1 align="center">Welcome to eyesight.org</h1>

<h6 align="right">(If you can read this, your eyesight is 20/20)</h6>
```

The `<p>` tag marks the beginning of a paragraph, leaving a blank line above it. It, too, can include a horizontal alignment attribute for center, left, or right alignment of the whole paragraph. In most cases, the closing `</p>` tag can be omitted, but it is safer to put it in, especially if the paragraph is not left-justified.

```
<h1>The Biggest Heading</h1>
<h2>A Smaller Heading</h2>
<!-- and so on >
<h6>The Smallest Heading</h6>
<p>One paragraph
<p>Next paragraph</p>
One line<br>
Next line
<blockquote>
Indented text
</blockquote>
<pre>Preformatted <u>text</u></pre>
<xmp>Preformatted <u>text</u> with no
interpretation of tags</xmp>
```

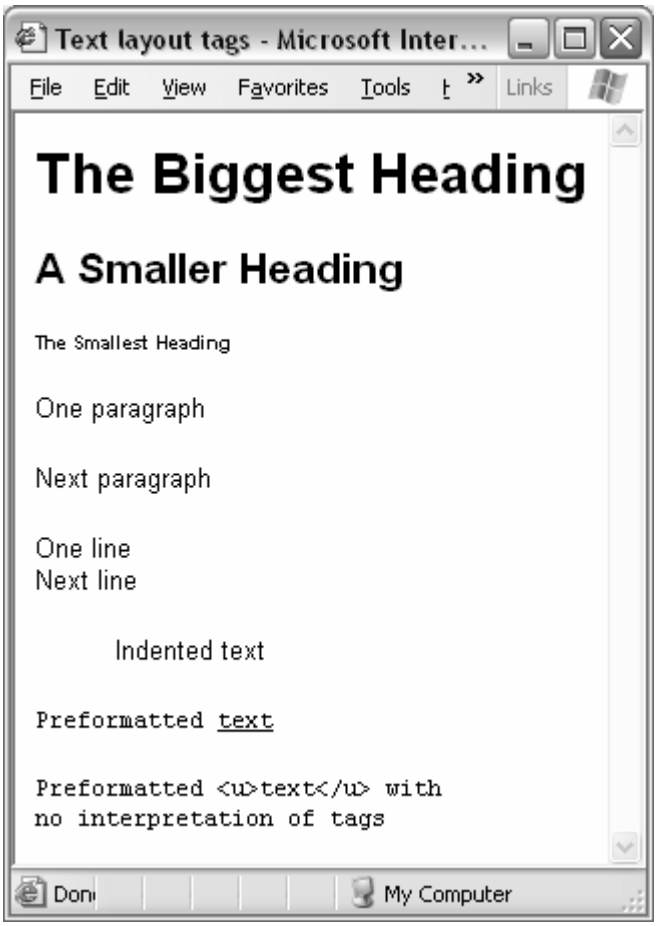


Figure C-4. Text layout tags

The
 tag marks the end of a line (in the displayed document, not in the source). For example, HTML source

```
<p>
An Internet poet named Braque
Had <br> tags all out of whack
```

will show up in a browser as:

> An Internet poet named Braque Had
> tags all out of whack

To display the lines properly, you need

```
<p>
An Internet poet named Braque<br>
Had &lt;br&gt; tags all out of whack<br>
```

Two
 tags in a row create a blank line.

We should also mention here the helpful "non-breaking space" character . Normally, browsers and HTML interpreters eliminate all white space from the document, replacing any series of line breaks and spaces in the source with one space in the displayed document. A non-breaking space inserts a space that does not disappear from the displayed document. A series of   ... leaves some empty space when displayed. If used between words,   keeps these words on the same line.

The <blockquote> tag indents all text between <blockquote> and </blockquote> and inserts blank lines above and below the indented paragraph. It is used for paragraph quotes (or for any indented text). <blockquote> tags can be nested.

The tag centers all the text and other elements (tables, images, etc.) until the closing tag.

The <hr> ("horizontal rule") tag makes a horizontal line. It may have the optional attributes width (length, usually in percents, relative to the width of the browser window), size (thickness), and color. For example:

```
<hr width="80%" size="3" color="green">
```

<pre> ... </pre> (preformatted text) and <xmp> ... </xmp> (sequence of literal characters) tags display the fragment of text between them using a fixed-width font and preserving all line breaks and spaces in the original text. This is useful for displaying fragments of computer programs, data files, and so on. The difference between the two is that <pre> interprets other HTML tags inside it. Therefore you have to use escape sequences in place of >, <, and & inside the <pre> tag. The <xmp> tag completely disables all HTML interpretation and shows the text exactly as it is in the source until </xmp>.

❖ ❖ ❖

The text attribute tags are summarized in Figure C-5. They provide for bold, italicized, and underlined text, smaller and bigger fonts, and subscripts and superscripts. The <code> tag enables a fixed-width font and is used to display computer-related items (for example, names of functions or variables in a computer program or HTML tags in an HTML tutorial). The <cite> tag is used for displaying names of books or articles. In fact it simply italicizes the text, just like the <i> tag, but it's better to use the tags according to their intended purpose. Similarly, the tag displays text in bold, just like <b>, but is more abstract, meaning highlighted text, while <b> just means bold. More recent HTML versions allow you to redefine the meanings of some tags using style sheets. For instance, may become red instead of bold.

```
<p> <b>Bold</b> <i>Italics</i> <u>Underlined</u>

<big>Big</big> <small>Small</small>

x<sub>1</sub> y<sup>2</sup>

<p> <code>Courier</code> <cite>Citation</cite>

<strong>Emphasis</strong>
```

Bold *Italics* Underlined Big Small x_1 y^2

`Courier` *Citation* **Emphasis**

Figure C-5. Text attribute tags

The <font> tag with the size attribute allows you to choose a font size from 1, the smallest, to 7, the largest. You can also increment or decrement the current size by preceding the value with a plus or minus sign. The default font size is 3. The color="*value*" attribute sets a different color for the font. For example,

```
<font size="+2" color="blue">
```

increments the current font size by two steps and sets the color to blue. Font attributes remain in effect until the closing `</font>` tag.

C.5 Anchors and Links

It is the `<a>` tag that converts ordinary text into hypertext. This tag is used in two ways: it can define an "anchor" or a link. An *anchor* simply marks a particular location in the document. To define an anchor, use the `<a>` tag with a `name` attribute. For example,

```
<a name="sectC_5">
<h2 align="center">Section C.5   Anchors and Links</h2>
```

The closing tag `</a>` is not required for an anchor.

Once an anchor is set, you can define hypertext *links* that send to this anchor. For example:

```
<p>The use of hypertext anchors and links is described
in <a href="#sectC_5">Section C.5</a>.
```

In this HTML fragment the `<a>` tag with an `href` attribute indicates that the words "Section C.5" are linked to the anchor `sectC_5`. A browser displays links in a color different from the rest of the text, and usually underlined. If you click on a link, the browser will take you to the position described in its `href` attribute. The # in front of `sectC_5` indicates that it is an anchor and not a file name.

In general, a link can take you to any *URL*. In Internet lingo, URL stands for *Uniform (or Universal) Resource Locator* and can represent a file (a web page) on the same computer (or as we say *host*) or any Internet address. If you write

```
<p>The use of hypertext links and anchors is described
in <a href="tutorial.html#sectC_5">Section C.5 of our tutorial</a>.
```

the link will take you to the file `tutorial.html`, presumably a different web page on the same web site as the current document, and position the display at the anchor `"sectC_5"` in it. If you omit the anchor, then the link will take you to the top of the other document. For example:

```
<p>The use of hypertext links and anchors is described
in <a href="tutorial.html">our tutorial</a>.
```

If your web site files are split between different directories, you should include the path for the file as part of the URL.

> **It is usually much better to use relative paths in URL links to documents on the same web site. A relative path defines the path going from the directory of the current document. A relative path does not begin with a slash.**
>
> **Use forward slashes in paths for compatibility with all systems.**

For example,

```
<p>The use of hypertext links and anchors is described
in <a href="courses/tutorial.html">our tutorial</a>.
```

The folder (directory) that holds the document containing the above lines has a subfolder, called `courses`, with the `tutorial.html` file in it. Here we go down one level in the directory tree. We can also go up one or several levels. For example:

```
<p align="center"><a href="../home.html">Back to the Home Page</a>
```

(`../` means go up one level, to the parent directory of the current directory.)

Finally, you can put into an `href` attribute the complete web address (URL) of a document on another web site. You can include an anchor in it if you wish. For example:

```
<p>A complete
<a href=
"http://www.html-helper.net/index.htm">
reference to HTML tags</a>
is available at html-helper.net.
```

This will appear in a browser as

A complete **reference to HTML tags** is available at html-helper.net.

Don't forget the `</a>` tag after your "hot" text, or the whole rest of your document may show up as one big hyperlink.

URLs do not have to point only to HTML documents: they can point to any type of file or resource. Your browser will decide what to do with a file or resource based on

its type. For example, it may call *Adobe Acrobat Reader* to display a PDF file, offer to download and save a zipped file on your computer's hard disk, or play music from an MP3 file. One particular URL that may be of interest has the form `mailto:` followed by an e-mail address. When you click on such a link, it brings up your e-mail program ready to send a message to the specified address. For example:

```
Report all problems with this web site to the
<a href="mailto:webmaster@nastyloops.com">webmaster</a>.
```

This will appear in a browser as

> Report all problems with this web site to the **webmaster**.

Note that URLs for your links are hidden on display and in printouts of your web page unless you repeat them explicitly in visible text. For example:

```
<p>An introduction to HTML tutorial can be
found at <a href="http://www.w3.org/MarkUp/Guide/">
<code>http://www.w3.org/MarkUp/Guide/</code></a>.
```

> An introduction to HTML tutorial can be found at **`http://www.w3.org/MarkUp/Guide/`**.

Or

```
The best way to reach me at any time of day or night is
to e-mail me at<br>
<a href="mailto:emailaddict@coffeeshop.org">
emailaddict@coffeeshop.org</a>.
```

> The best way to reach me at any time of day or night is to e-mail me at **emailaddict@coffeeshop.org**.

C.6 Lists

HTML lets you put a list of items on your page. It supports three types of lists: an unordered (bulleted) list using the <ul> tag, an ordered (numbered) list using the <ol> tag, and a "definition list" of items with captions using the <dl> tag. These lists are illustrated in Figure C-6.

```
<ul>
<li>Item 1</li>
<li>Item 2</li>
<li>Item 3</li>
</ul>

<ol>
<li>Item 1</li>
<li>Item 2</li>
<li>Item 3</li>
</ol>

<dl>
<dt>Item 1</dt>
<dd>Definition 1</dd>
<dt>Item 2</dt>
<dd>Definition 2</dd>
<dt>Item 3</dt>
<dd>Definition 3</dd>
</dl>
```

- Item 1
- Item 2
- Item 3

1. Item 1
2. Item 2
3. Item 3

Item 1
    Definition 1
Item 2
    Definition 2
Item 3
    Definition 3

Figure C-6. , <ol>, and <dl> tags for lists

For the <ul> tag, you can add the type attribute with the possible values "circle", "disc", or "square" which defines the shape of the bullet ('disk" is the default). In the <ol> tag, the type attribute may have the values "1" (default), "I", "i", "A", and "a", and it defines the style of numbers (regular numbers, upper or lower case Roman numerals, or upper or lower case letters). You can also change the starting number with the start="*value*" attribute. In both unordered and ordered lists each item is placed between <li> ("list item") and tags. In a "definition list," the term is placed between <dt> and </dt> tags ("definition term") and the definition itself is placed between <dd> and </dd> tags ("definition description").

HTML lets you nest one list inside another (Figure C-7). In this example we used indentation and blank lines in the source to make the source more readable. They do not affect the displayed page.

```
<h4 align="center">Java Programming Style</h4>

<ol>
<li>White space
  <ol type="A">
  <li>Use spaces around all operators,
  to the left of <code>(</code> and <code>{</code>
  and to the right of <code>)</code> and <code>}</code></li>
  <li>Use blank lines to separate meaningful segments
  of code</li>
  </ol>
</li>

<li>Names
  <ol type="A">
  <li>Use meaningful names for classes, methods and
  variables</li>
  <li>Use nouns for classes and verbs for methods</li>
  </ol>
</li>
</ol>
```

Java Programming Style

1. White space
   A. Use spaces around all operators, to the left of (and { and to the right of) and }
   B. Use blank lines to separate meaningful segments of code
2. Names
   A. Use meaningful names for classes, methods and variables
   B. Use nouns for classes and verbs for methods

Figure C-7. Nested lists

C.7 Images

One picture is worth a thousand HTML tags. The two most common formats for images on the Internet are *GIF* (Graphics Interchange Format) and *JPEG* (Joint Photographic Experts Group) format. GIF file names have the extension `.gif` and JPEG files have the extension `.jpg`. Both use image compression because uncompressed image files can be quite large.

Images are incorporated into an HTML document using the <img> tag. This tag has many attributes of which the most important one, and the only one required, is the source of the image. It has the form src="*url*", where *url* is a name of a file, usually somewhere on the same web site. For example,

```
<img src="wintertrack.jpg">
```

Other attributes are:

- align with values "top", "center", or "bottom". Specifies vertical alignment of the image with respect to text in the same line and is useful for small images: icons, bullets, and so on.
- alt="*text*". Specifies the text to display in place of the image when the image is not available for some reason. Some browsers also show this text inside a yellow label that appears when you position the mouse cursor over the image.
- border="*thickness*". Specifies the thickness of the border. 0 means no border.
- height="*value*" and width="*value*". Specify the dimensions of the image in pixels.

Height and width attributes are optional, but they allow the browser to reserve space for the image and continue loading the page while the image file is still on its way. If the specified dimensions are different from the actual image dimensions, the image is scaled, which may result in poor quality.

A more complete <img> tag with attributes may look like

```
<img src="wintertrack.jpg" width="320" height="100" border="0"
       alt="Girls winter track team">
```

There is no closing tag for <img>.

You can surround an image tag with ... tags. Then the whole picture becomes a "hot" button that, when touched, will take the user to the specified URL.

❖ ❖ ❖

A more interesting trick is to make different areas in a picture active with different URLs attached to them. You can do this using a <map> tag with a list of areas in it. Each area can have its own URL, which is activated when that area is touched. The

areas may include circles, rectangles, and polygons; you can define more complex areas by using several overlapping shapes with the same URL. An image can be tied to a particular map by adding a `usemap="#mapname"` attribute to the `<img>` tag. For example:

```
<img src="wintertrack.jpg" width="320" height="100" border="0"
        alt="Girls winter track team" usemap="#teammap">

<map name="teammap">
  <area shape="circle" coords="10,20,30"   href="Lucy.jpg">
  <area shape="rect" coords="60,20,85,40"  href="Sean.jpg">
  <area shape="polygon" coords="25,150,50,100,100,100,150,200"
                                           href="Melissa.html">
</map>
```

In this example, the map “teammap” overlaps the picture. Presumably areas in the map outline faces in the photograph and when you click on an area, the browser opens a page with a description or a bigger picture of that person. The coordinates are in pixels, with the origin in the upper-left corner of the image. A circle is described by the *x* and *y* coordinates of its center and its radius. A rectangle is described by the coordinates of its two opposite corners. A polygon is described by the coordinates of its vertices.

C.8 Tables

The `<table>` ... `</table>` tags define a table in your document. You have to describe each row of the table separately using the `<tr>` ... `</tr>` (“table row”) tags. Within each row you describe each individual table entry between the `<td>` ... `</td>` (“table data”) tags.

Figure C-8 shows HTML example of a table used to display a calendar for a month.

This calendar will look rather ugly (Figure C-9-a): the day names in the heading row do not stand out and the numbers are not right-justified. The first problem is easy to fix: all you have to do is replace all the `<td>` tags with `<th>` (“table heading”) tags in the first row:

```
<tr>
<th>Sun</th><th>Mon</th><th>Tue</th><th>Wed</th><th>Thu</th>
<th>Fri</th><th>Sat</th>
</tr>
```

```
<p><b>November 2001</b></p>

<p>
<table>
<tr>
<td>Sun</td><td>Mon</td><td>Tue</td><td>Wed</td><td>Thu</td>
<td>Fri</td><td>Sat</td>
</tr>

<tr>
<td></td><td></td><td></td><td></td><td></td><td>1</td><td>2</td><td>3</td>
</tr>

<tr>
<td>4</td><td>5</td><td>6</td><td>7</td><td>8</td>
<td>9</td><td>10</td>
</tr>

<tr>
<td>11</td><td>12</td><td>13</td><td>14</td><td>15</td>
<td>16</td><td>17</td>
</tr>

<tr>
<td>18</td><td>19</td><td>20</td><td>21</td><td>22</td>
<td>23</td><td>24</td>
</tr>

<tr>
<td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td>
</tr>

</table>
```

Figure C-8. HTML document that uses a table to show a calendar

To fix the second problem add the `align="right"` attribute to each `<tr>` tag:

```
<tr align="right">
<th>Sun</th><th>Mon</th><th>Tue</th><th>Wed</th><th>Thu</th>
<th>Fri</th><th>Sat</th>
</tr>
 ... etc.
```

(a)

November 2007

Sun	Mon	Tue	Wed	Thu	Fri	Sat
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

(b)

November 2007

Sun	Mon	Tue	Wed	Thu	Fri	Sat
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

(c)

November 2007

Sun	Mon	Tue	Wed	Thu	Fri	Sat
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	**22**	23	24
25	26	27	28	29	30	

Figure C-9. Formatting tables in HTML

You can use the `align` attribute with individual `<td>` and `<th>` tags; then they will override the alignment specified in `<tr>`.

With these changes the result looks a little better (Figure C-9-b), but the column widths are still uneven and the heading is too close to the numbers. You can add an empty row after the first row to separate the day names from the numbers. To make it show up, put one non-breaking space in it. For example:

```
<tr><td> </td></tr>
```

To make the columns even, you can add the `cols="7"` and `width="300"` attributes to the `<table>` tag. These attributes set the number of columns and table's width in pixels. Unfortunately, not all browsers seem to understand the `cols` attribute. It may be safer to specify the width of each column by adding the `width` attribute to each `<td>` tag (at least in the first row of the table).

You can also specify the width of the table relative to the width of the browser window. For example: `width="30%"`. If you specify column widths for individual columns with percent values, the column widths will be set relative to the width of the whole table.

For a finishing touch, we might mark Thanksgiving on the 22nd (Figure C-9-c). Let's make it "lightcoral" on a "lavenderblush" background (these color names come from the list of HTML-supported colors [1]):

```
<td bgcolor="lavenderblush"><font color="lightcoral">22</font></td>
```

Working with tables is not easy because `<table>`, `<tr>`, and `<td>` tags have many attributes and `<td>` attributes can override the same attributes for the `<tr>` tag. On top of this, different browsers or versions of the same browser may support and interpret some of the attributes differently. So you have to be careful and test your page on different platforms. If your web page is for everyone, you should try to use safer proven methods.

On the other hand, tables are very powerful. HTML does not give you many tools for controlling page layout: tables are pretty much all you have. You can use tables for all kinds of things: side bars or horizontal bars with navigation links, narrow columns of text and multi-column text, text and picture insets, blocks of text with borders or color backgrounds, and so on. Lists, images, and applets may all be placed inside a table cell, and you can have tables within tables. Look at some web sites you like for examples of how tables are used.

The following attributes will add flexibility to your tables:

`<table>`
- `border` — border thickness in pixels; 0 means no border
- `cellspacing` — space between a table's cells
- `cellpadding` — additional space between data and the borders of the cell
- `hspace` and `vspace` — additional horizontal and vertical space around the table

`<tr>`, `<th>`, and `<td>`
- `align` (`"left"`, `"center"`, or `"right"`) — horizontal alignment
- `valign` (`"top"`, `"middle"`, or `"bottom"`) — vertical alignment (inside the cell)
- `bgcolor` — background color

The `rowspan` and `colspan` attributes help create fancier tables, in which a cell can span several rows or columns [1].

You can also place a `<caption>` tag inside a table definition (but outside of all row definitions):

```
<caption align="..."> ... </caption>
```

The `align` attribute here can be `"top"` (top of the table, default) or `"bottom"`.

C.9 Applets

Finally, you can add an *applet*, a little Java program, to a web page. This is done using an `<applet>` tag.

> **Applet tags can appear in more or less any place in your HTML document; a web page can run several applets or several copies of the same applet concurrently.**

The following example loads and runs the *TicTacToe* applet from the standard Java demo collection and provides a link to its source code, `TicTacToe.java`:

```
<html>
  <head>
    <title>TicTacToe v1.1</title>
    </head>
  <body>
      <h1>TicTacToe v1.1</h1>
      <hr>
      <applet code=TicTacToe.class width=120 height=120>
          alt="Your browser understands the &lt;APPLET&gt;
                tag but isn't running the applet, for some reason."
          Your browser is completely ignoring the &lt;APPLET&gt; tag!
      </applet>
      <hr>
      <a href="TicTacToe.java">The source.</a>
  </body>
</html>
```


The `code` attribute defines the name of the applet's `class` file, `TicTacToe.class` in this example.

> **Applet class file names are case-sensitive.**

The `codebase` attribute specifies a path to the directory (folder) that contains the applet's code. It can be the absolute path but more often it is the path relative to the

location of your HTML file. You do not need to specify `codebase` at all (or you can put `codebase="."` for the current directory) if the applet's code is in the same folder as your HTML source. `codebase` may be case-sensitive, too, depending on the operating system. In the above example, the `TicTacToe.class` file happens to be in a folder called `TicTacToe` somewhere under JDK's demos.

`width` and `height` attributes define the dimensions of the window in which you want the applet to run. More often than not these dimensions depend on the applet itself: few applets are written to be completely scaleable. Try to open the *TicTacToe* applet and play it with *width*=50 and *height*=50, and you'll see what happens.

You can use the `align` attribute with a value of `"left"` or `"right"` to align your applet with the left or the right margin of the web page. You can also use it with a value of `"top"`, `"middle"`, or `"bottom"` to place the applet within the current text line and align it with the top, middle, or bottom of the text.

The `alt` attribute tells the browser what message to display if the applet fails to run for whatever reason. You can also put any text between the `<applet>` and `</applet>` tags; this text will be visible only if your browser does not recognize the `<applet>` tag at all.

Occasionally an applet needs one or several parameters to run. You can set the values of parameters using `<param>` tags placed between `<applet>` and `</applet>`. The format for the `<param>` tag is:

```
<param name="anyName" value="anyText">
```

anyName is the parameter's name, defined by the programmer. It is used in the program to fetch the parameter's value by calling `JApplet`'s `getParameter` method. The value is a string of text; the program can convert it into a number or extract several words or numbers from it.

Exercises

Sections C.1 - C.4

1. Which of the following are valid HTML tags?

`<u>`, `<small>`, `<sup>`, `<red>`, `<strong>`

2. Choose one of the tags, `<pre>`, `<cite>`, `<blockquote>`, `<xmp>`, or `<hr>`, to accomplish each of the following tasks:

(a) Indent all lines between the opening and closing tags.
(b) Put all text between the opening and closing tags in italics.
(c) Stop processing all HTML tags between the opening and closing tags.
(d) Display a horizontal line.
(e) Display the text between the opening and closing tags in a fixed-width font and show spaces and line breaks exactly as in the source, but apply formatting from HTML tags embedded inside.

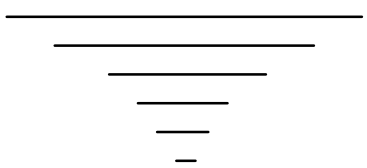
3. (a) Create an HTML document with the phrase:

HTML stands for **H**yper**T**ext **M**arkup **L**anguage.

formatted as shown above.

(b) Make the background color blue and set the default text color to white for the whole document.

4. Show the following design on your web page using the `<hr>` tags:



5. Create an HTML document that looks in a browser window as shown inside the box below. The first paragraph is text that fills the width of the browser window. It is automatically split into lines as necessary when the window is resized. The program segment should look exactly as shown.

Console Output in Java

The `System.out` object is defined in each Java program. It has methods for displaying text strings and numbers in plain text format on the system display. For example:

```
public class Hello
{
  public static void main(String[] args)
  {
    System.out.println("Hello, World!");
  }
}
```

⁜ Hint: do not use `<br>` tags in the first paragraph — let the browser fill the lines. ⁜

6. A limerick is a five-line poem (usually funny or silly) with a particular rhythm and rhyming pattern [1]. Finish the limerick on page 12 and put it into an HTML document, formatted as follows:

An Internet poet named Braque
Had
 tags all out of whack

By ---

For example:

An Internet poet named Braque
Had
 tags all out of whack
His non-breaking spaces
Would get out of places
But nested lists just blew his stack!

By Brenda Webb

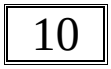
7. What happens when `<xmp>` ... `</xmp>` tags are embedded inside `<pre>` ... `</pre>` tags? Experiment with different browsers and report the results.

Sections C.5 - C.9

8. The `<ul>` tag marks a bulleted list. Create your own bulleted list (for example, a list of things to do) without using the `<ul>` tag. Find a collection of free icons on the web and choose one for your bullet. Then create a bulleted list using the `<blockquote>`, `<img>`, and `<br>` tags and, if necessary, non-breaking space characters.

9. The `<ul>` ... `</ul>` or `<ol>` ... `</ol>` tags (without any `<li>` tags) can be used to indent text. Explore the differences between one `<ul>`, two `<ul>` tags next to each other, one `<blockquote>`, and two `<blockquote>` tags next to each other.

10. Come up with a way to show a boxed number on your web page. For example:



11. Create an HTML document that shows a partial list of Java reserved words, arranged in a table with four columns (see page 107).

12.■ Create a web page that shows a chessboard with the solution to the *Eight Queens* problem using only text — no images. ⦇ Hint: use a table and set alternating background colors for its cells. ⦈ Add to your page links to a few web pages that describe the problem.

13. (a) Create a web page with a list of people from your class.
(b) Create a clickable e-mail directory for your class.

14.■ Create a web page with text in the middle column and pictures to the left and to the right of it. The text should occupy 1/3 of the browser window width. The pictures should be centered vertically with respect to the text.

15.■ Create a picture gallery for your friends, family, town, team, or band, or for your pet. Use a list or a table of clickable reduced pictures as an index for your gallery. You will need two versions of each picture: a large one and a reduced one. (Reduced pictures can be scanned at different resolution or created using any image editing program, such as Adobe's *Photoshop*.)

16.◆ Create a personal web site consisting of at least two pages (HTML documents) with links among them. Use headings, images and tables.