



**University of Babylon**

**College of Information Technology**

**Department of Information Networks**

# DYNAMIC HTML

Lecturer

*Haider M. Habeeb*

Second Year, First Course

2012 - 2013

## The Internet

The Internet is simply a catchall phrase referring to the vast, globe-spanning network of computers that are connected to each other and are able to transmit and receive data, shuttling information back and forth around the world at nearly the speed of light. It has been around in some form for almost half a century now, ever since a few very smart people figured out how to make one computer talk to another computer. The Internet has since become so ubiquitous and pervasive, impacting so many aspects of modern life, that it's hard to imagine a world without it.

## The World Wide Web

The World Wide Web is just one facet of the Internet, like a bustling neighborhood in a much larger city. It's made up of millions of files and documents residing on different computers across the Internet, all cross-referenced and interconnected to weave a web of information around the world, which is how it gets its name. In its relatively short history, the web has grown and evolved far beyond the simple text documents it began with, carrying other types of information through the same channels: images, video, audio, and fully immersive interactive experiences. But at its core, the web is fundamentally a text-based medium, and that text is usually encoded in HTML.

Many different devices can access the web: desktop and laptop computers, personal digital assistants (PDAs), mobile phones, game consoles, and even some household appliances. Whatever the device, it in turn operates software that has been programmed to interpret HTML. These programs are technically known as user-agents, but the more familiar term is web browsers.

## The Web Browser

A web browser is specifically a program intended to visually render web documents, whereas some user-agents interpret HTML but don't display it.

A browser or user-agent is also known as a client, because it is the thing requesting and receiving service. The computer that serves data to the client is, not surprisingly, known as a server. The Internet is riddled with servers, all storing and processing data and delivering it in response to client requests. The client and the server are two ends of the chain, connected to each other through the Internet.

## The Effect of Browser Wars

Browser incompatibility is the main cause for the necessity to have different methods to access layers. When Netscape Navigator 4 was introduced, it was the first "DHTML browser." Unfortunately, it was released before the World Wide Web Consortium could issue recommended standards for the DOM.

Previous problems with coding DHTML technologies occurred because the browser manufacturers created their own non-standardized, proprietary features specific to the use of that particular browser. This has resulted in Web pages that look fine on one browser, but not with other browsers. Microsoft Internet Explorer 4, though not perfect used a more advanced DOM where every HTML element is programmable. Since then, we have seen the introduction of W3C standards, Internet Explorer 5 and 6, Netscape 6 and 7, and Firefox. DHTML has become a much more powerful tool and, more importantly, a standard.

## Dynamic HTML

DHTML is the art of combining HTML, JavaScript, DOM, and CSS.

DHTML is NOT a Language

DHTML stands for **D**ynamic **H**yper**T**ext **M**arkup **L**anguage.

DHTML is NOT a language or a web standard.

DHTML is a combination of HTML, JavaScript, DOM and CSS.

According to the World Wide Web Consortium (W3C):

*"Dynamic HTML is a term used by some vendors to describe the combination of HTML, style sheets and scripts that allows documents to be animated."*

## HTML

The W3C HTML 4 standard has rich support for dynamic content:

- HTML supports JavaScript
- HTML supports the Document Object Model (**DOM**)
- HTML supports HTML Events
- HTML supports Cascading Style Sheets (**CSS**)

## JavaScript

JavaScript is the most popular scripting language on the internet, and it works in all major browsers.

DHTML uses JavaScript to control, access, and manipulate HTML elements.

## HTML DOM

The HTML DOM is a W3C standard. It describes the **Document Object Model** for HTML.

The HTML DOM defines a standard way for accessing and manipulating HTML documents.

DOM is simply a framework containing objects displayed by a browser, such as forms, images, layers, etc. Developers can employ this hierarchical structure to access and manipulate what appears onscreen.

## HTML Events

HTML events are a part of the HTML DOM.

DHTML is about creating web pages that reacts to (user) events.

## CSS

CSS defines how to display HTML elements.

DHTML is about using JavaScript and the HTML DOM to change the style and positioning of HTML elements.

## **Static vs. Dynamic HTML**

Until the Version 4 browsers were released, we only knew static HTML. That means: we put HTML elements (paragraphs, images, etc.) in a specific order in the source code. The browser always showed all elements in this order. Positioning and styling was done by tables, div's and such aids. If we wanted to change the order or the positioning of the elements, we had to rewrite the HTML.

Dynamic HTML gives us the chance to re-organize our pages on the fly. In fact, we can take some elements out of the natural flow of the page, put them somewhere else and change its position again if the user clicks a link. The natural flow of the page is the page as the browser first shows it. It calculates the tables and paragraphs and other things we put in the page, then displays them in the best possible way, one by one, from the beginning to the end of the HTML document.

## **How does DHTML work?**

Dynamic HTML gives authors creative control so they can manipulate any page element and change styles, positioning, and content at any time. It provides a richer, more dynamic experience on web pages, making them more like dynamic applications and less like static content. Dynamic HTML presents richly formatted pages and lets you interact with the content on those pages without having to download additional content from the server. This means that a page can respond immediately to user actions, such as a mouse click, without having to retrieve an entire new page from the server.

## HTML Basics

### What is HTML?

HTML is a language for describing web pages.

- HTML stands for **H**yper **T**ext **M**arkup **L**anguage
- HTML is not a programming language, it is a markup language
- A markup language is a set of markup tags
- HTML uses markup tags to describe web pages

### Understand HTML

HTML stands for Hypertext Markup Language, and it is the language in which virtually all Web pages are written. Now, don't break out in hives when you hear the word "language." You don't need complex logical or mathematical formulas to work with HTML, and you don't need to think like a programmer to use it. Computer programmers must think through the tasks that they want their programs to perform, and then develop an elaborate (and usually complicated) series of instructions to tell the computer what to do. Although you do need to do some thinking and planning when you use HTML, it is not nearly that difficult. So, how does Hypertext Markup Language work?

Hypertext refers to the way in which Web pages (HTML documents) are linked together. When you click a link in a Web page, you are using hypertext. It is this system of linking documents that has made the World Wide Web the global phenomenon it has become.

Markup Language describes how HTML works. With a markup language, you simply "mark up" a text document with tags that tell a Web browser how to structure and display it. HTML originally was developed with the intent of defining the structure of documents (headings, paragraphs, lists, and so forth) to facilitate the sharing of scientific information between researchers. All you need to do to use HTML is to learn what type of markup to use to get the results you want.

### Markup

The first step toward understanding and working with HTML is learning the basic terms that describe most of the functions of this language. You will come across these terms repeatedly as you use HTML and, if you understand them, you will have progressed a long way toward comprehending HTML itself.

### Elements

All HTML pages are made up of elements. Think of an element as a container in which a portion of a page is placed. Whatever is contained inside the element will take on the characteristics of that element. For example, if you want to put a large heading on a page, you would enclose it in a heading element `<h1> </h1>`. If you want to create a table, you put the table information inside the table element `<table> </table>`. To construct a form, you need the form element `<form> </form>`.

### Tags

Often, you'll find the terms element and tag used interchangeably. It's fairly common, but not strictly accurate. An element is made up of two tags: an opening tag and a closing tag. Although it might seem somewhat picky to make this distinction, when XML (eXtensible markup language) becomes the big boy of the Web, it will be a very

important difference to remember. So, get in the habit of distinguishing between elements and tags, and you'll save yourself some confusion down the line.

All tags are constructed the same way. The tag begins with a "less than" sign (<), then the element name, followed by a "greater than" sign (>). For example, an opening tag for the paragraph element would look like this: <p>. The only difference in a closing tag is that the closing tag includes a slash (/) before the element name: </p>. Your content goes between the tags. A simple paragraph might look like this:

```
<p>This is an HTML paragraph.</p>
```

## Empty elements

Some HTML elements do not use closing tags. These are called empty elements. For example, the line break element <br> does not require a closing tag. Whereas HTML is flexible where some closing tags are concerned, XHTML and XML are not. The best way to be prepared for the future is by getting in the habit of always using opening and closing tags. In the case of empty elements, write them like this: <br />. When a browser sees the slash, it will recognize the element as one that does not need a separate, closing tag.

## Understand Document Elements

The elements listed in the following table could be called document elements because they describe and define your HTML document. These are some of the document elements that you will use them right now. The rest of the elements will study them at suited time later during this course.

Element	Description
<html> </html>	Defines the beginning and end of the document.
<head> </head>	This is the document header. It works like a storehouse of important information about the document. This information generally is not displayed in the document.
<title> </title>	The title element is nested inside the header. It displays a page title in the title bar at the top of the browser.
<body> </body>	The main body of the Web page goes inside this element.

## Create an HTML Template

For making your work easy it's a good to create an HTML template. It's simply a file that has all the basic elements for a Web page already written.

To create an HTML template, follow these steps:

1. Open Notepad or another text editor.
2. In the File menu, choose Save As.
3. Save the file as template.html.
4. At the top of the page type <html>.
5. Now add the header element: <head> </head>.
6. Inside the header, type <title> </title>.
7. On the next line, type <body> </body>.
8. Finally, type </html>.
9. Click the Save icon.

You have created a template that you can use whenever you want to make a new Web page. Your HTML code should look something like this:

```
<html>
<head><title></title></head>
<body></body>
</html>
```

Or you can arrange it like this:

```
<html>
<head>
<title>
</title>
</head>
<body>
</body>
</html>
```

## HTML Headings

The purpose of the heading element is to indicate different heading levels in a document. The tags are made up of an h with a number following it. For example, to specify a level one heading, you would write:

```
<h1>This is a level one heading.</h1>
```

A level two heading would look like this:

```
<h2>This is a level two heading.</h2>
```

HTML includes six heading levels: <h1>, <h2>, <h3>, <h4>, <h5>, and <h6>. Although the primary function of the <h#> element is to specify headings, it also often is used as a quick way to tell a Web browser to display different sizes of text. Try typing the following HTML code to see how the six heading levels will display on a Web browser:

```
<html>
<head> <title> Headings </title></head>
<body>
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
<h4>This is heading 4</h4>
<h5>This is heading 5</h5>
<h6>This is heading 6</h6>
</body>
</html>
```

## HTML Paragraphs

HTML paragraphs are defined with the <p> tag. Write the following HTML document and see how it looks in the browser.

```
<html>
<head> <title> Paragraphs </title></head>
<body>
<p>This is a paragraph.</p>
<p>
This is another paragraph.
</p>
</body>
</html>
```

Let us write a long paragraph with multiline and see the result in the browser.

```
<html>
<head> <title> Paragraphs </title></head>
<body>
<p>
Dynamic HTML is a combination of:
1- HTML
2- CSS
3- DOM
4- JavaScript
</p>
</body>
</html>
```

The result in the browser does not look as we expect. It appears on one line like this:

Dynamic HTML is a combination of:1- HTML2- CSS3- DOM4- JavaScript

So, we do understand that we need something help us arrange the paragraphs.

### Create Line Breaks with **<br />**

**<br />** is an empty element (doesn't have end tag) that's why we have to write it in this way: "<" then "br" then space then ">".

The function of this element is to insert line break and go to the next line. While **<p>** will add new line before starts your paragraph.

Then, we can modify the previous HTML document to show the perfect result:

```
<html>
<head> <title> Paragraphs </title></head>
<body>
<p>
Dynamic HTML is a combination of:<br />
1- HTML<br />
2- CSS<br />
3- DOM<br />
4- JavaScript
</p>
</body>
</html>
```

### Control Text with Character Elements

The character elements give you some basic controls over how text will display on a Web browser. As you work with these elements, you will notice that some of them are named for how a character actually will look when it appears on a browser (for example, the bold **<b>** element for bold text). These are called physical elements because they describe the physical appearance of the characters. Other text elements derive their names from the intended function or purpose of the text (for example, the **<strong>** element also will display bold typeface, but the element's name stands for strongly emphasized text). When a character element is named this way, it is called a logical element because it describes the logical function of the text.

```
<html>
<head> <title> Formatting Text </title></head>
<body>
<p><b>This text is bold</b></p>
<p><strong>This text is strong</strong></p>
<p><big>This text is big</big></p>
<p><i>This text is italic</i></p>
<p><small>This text is small</small></p>
<p>This is<sub> subscript</sub> and <sup>superscript</sup></p>
</body>
</html>
```