COMPUTER NETWORKS

Lecture Notes

Course Code - BCS 415

Course Name - INTERNET & WEB TECHNOLOGY-II (3-1-0) Cr.-4

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING, IT

Veer Surendra Sai University of Technology

Burla-768018

BCS-415 **INTERNET & WEB TECHNOLOGY-II (3-1-0)Cr.-4**

TCP/ IP:

TCP/IP Overview, TCP/IP and Internet, Layers of TCP/IP, Network Layer: Addressing, Sub netting, concepts of ARP, RARP, ICMP, IGMP. Transport Layer: UDP & TCP, Application Layer: Client server model, BOOTP, DHCP, DNS, TELNET, FTP, SMTP model, HTTP, idea of WWW and CGI.

Core JAVA:

JAVA fundamentals, overview of JAVA operators, control statements, introducing classes, inheritance, exception handling, AWT, working with window graphics and text, AWT controls, Layout manager.

Advanced JAVA:

Introducing threading, advantages, Multi-threading, JAVA and networking, TCP/IP client sockets, Whois, URL, Server sockets, Overview of a caching Proxy HTTP server.

Applets and JDBC:

Introducing Applets, Architecture of an applet, skeleton, HTML APPLET tag, Event Handling, JDBC, Connecting to a database, transactions and executing SQL query, JDBC interface, Callable and prepared statements, Introduction to swing.

CGI Programming

Introduction to Web Architecture, Apache Web Server, Perl programming, CGI programming with Perl.

Network Security:

Network security basics and needs, cryptography, encryption and decryption, Ciphertext, types of cryptography: symmetric and asymmetric, RSA algorithm, Digital Signature, Organizational security issues and firewall architecture.

Reference Books:

1. TCP/IP – Forouzan (TMH)

2. Internet and World Wide Web, How to Program, Dietel and Dietel, Pearson Education.

3. Complete Reference JAVA - Naughton Schildt

4. Web Technologies – Achyut S Godbole and Atul Kahate

INTERNET AND WEB TECHNOLOGY

Understanding the WWW and the Internet:

Internet: The Internet is a global system of interconnected computer networks that use the standard Internet Protocol Suite (TCP/IP) to serve billions of users worldwide. It is a *network of networks* that consists of millions of private, public academic, business, and government networks.

WWW: The World Wide Web, abbreviated as WWW and commonly known as the Web, is a system of interlinked hypertext documents accessed via the Internet. With a web browser, one can view web pages that may contain text, images, videos, and other multimedia and navigate between them via hyperlinks.

Emergence of Web: Between the summers of 1991 and 1994, the load on the first Web server ("info.cern.ch") rose steadily by a factor of 10 every year.

In 1992 academia, and in 1993 industry, was taking notice. World Wide Web Consortium is formed in September 1994, with a base at MIT is the USA, INRIA in France, and now also at Keio University in Japan.

With the dramatic flood of rich material of all kinds onto the Web in the 1990s, the first part of the dream is largely realized, although still very few people in practice have access to intuitive hypertext creation tools.

The second part has yet to happen, but there are signs and plans which make us confident. The great need for information about information, to help us categorize, sort, pay for own information is driving the design of languages for the web designed for processing by machines, rather than people. The web of human readable document is being merged with a web of machine-understandable data. The potential of the mixture of humans and machines working together and communicating through the web could be immense.

WEB Servers: To view and browse pages on the Web, all you need is a web browser. To publish pages on the Web, you need a web server. A web server is the program that runs on a computer and is responsible for replying to web browser requests for files. You need a web server to publish documents on the Web. When you use a browser to request a page on a website, that browser makes a web connection to a server using the HTTP protocol. The browser then formats the information it got from the server. Server accepts the connection, sends the contents of the requested files and then closes.

WEB Browsers:

A web browser is the program you use to view pages and navigate the World Wide Web. A wide array of web browsers is available for just about every platform you can imagine. Microsoft Internet Explorer, for example, is included with Windows and Safari is included with Mac OS X. Mozilla Firefox, Netscape Navigator, and Opera are all available for free.

What the Browser Does The core purpose of a web browser is to connect to web servers, request documents, and then properly format and display those documents. Web browsers can also display files on your local computer, download files that are not meant to be displayed. Each web page is a file written in a language called the Hypertext Markup

Language (HTML) that includes the text of the page, a description of its structure, and links to other documents, images, or other media.

Protocols: In computing, a protocol is a set of rules which is used by computers to communicate with each other across a network. A protocol is a convention or standard that controls or enables the connection, communication, and data transfer between computing endpoints.

Internet Protocol Suite: The Internet Protocol Suite is the set of communications protocols used for the Internet and other similar networks. It is commonly also known as TCP/IP named from two of the most important protocols in it: The Transmission Control Protocol (TCP) and the Internet Protocol (IP), which were the first two networking protocols defined in this standard.

Building Web sites: It's a good idea to first think about and design your site. That way, you'll give yourself direction and you'll need to reorganize less later.

To design your site:

1. Figure out why you're creating this site. What do you want to convey?
2. Think about your audience. How can you tailor your content to appeal to this audience? For example, should you add lots of graphics or is it more important that your page download quickly?
3. How many pages will you need? What sort of structure would you like it to have? Do you want visitors to go through your site in a particular direction, or do you want to make it easy for them to explore in any direction?
4. Sketch out your site on paper.

Devise a simple, consistent naming system for your pages, images and other external files.

HTML

Planning for designing web pages:

Breaking Up Your Content into Main Topics

With your goals in mind, try to organize your content into main topics or sections, chunking related information together under a single topic.

Ideas for Organization and Navigation

At this point, you should have a good idea of what you want to talk about as well as a list of topics. The next step is to actually start structuring the information you have into a set of web pages. Before you do that, however, consider some standard structures that have been used in other help systems and online tools. This section describes some of these structures, their various features, some important considerations, including the following

Model and Structure of a Web site:

You need to know what the following terms mean and how they apply to the body of work you're developing for the Web:

Website: A collection of one or more web pages linked together in a meaningful way that, as a whole, describes a body of information or creates an overall effect.

Web server: A computer on the Internet or an intranet that delivers Web pages and other files in response to browser requests.

Web page: A single document on a website, usually consisting of an HTML document and any items that are displayed within that document such as inline images.

Home page: The entry page for a website, which can link to additional pages on the same website or pages on other sites.

Developing websites:

Designing a website, like designing a book outline, a building plan, or a painting, can sometimes be a complex and involved process. Having a plan before you begin can help you keep the details straight and help you develop the finished product with fewer false starts. Today, you learned how to put together a simple plan and structure for creating a set of web pages, including the following:

• Deciding what sort of content to present
• Coming up with a set of goals for that content
• Deciding on a set of topics
• Organizing and storyboarding the website

Basic HTML: HTML stands for Hypertext Markup Language. The idea here is that most documents have common elements for example, titles, paragraphs, and lists. Before you start writing, therefore, you can identify and define the set of elements in that document and give them appropriate names.

How Markup Works

HTML is a markup language. Writing in a markup language means that you start with the text of your page and add special tags around words and paragraphs. The tags indicate the different parts of the page and produce different effects in the browser. HTML has a defined set of tags you can use. You can't make up your own tags to create new styles or features.

What HTML Files Look Like

Pages written in HTML are plain text files (ASCII), which means that they contain no platform- or program-specific information. Any editor that supports text can read them. HTML files contain the following:

• The text of the page itself
• HTML tags that indicate page elements, structure, formatting, and hypertext links to

other pages or to included media. Most HTML tags look something like the following:
<thetagname>affected text</thetagname>
The tag name itself (here, thetagname) is enclosed in brackets (< >). HTML tags generally have a beginning and an ending tag surrounding the text they affect. The beginning tag "turns on" a feature (such as headings, bold, and so on), and the ending tag turns it off. Closing tags have the tag name preceded by a slash (/). The opening tag (for example, for paragraphs) and closing tag (for example, for paragraphs) compose what is officially called an HTML element.

Text Formatting and HTML

When an HTML page is parsed by a browser, any formatting you might have done by hand that is, any extra spaces, tabs, returns, and so on is ignored. The only thing that specifies formatting in an HTML page is an HTML tag. If you spend hours carefully editing a plain text file to have nicely formatted paragraphs and columns of numbers but don't include any tags, when a web browser loads the page, all the text will flow into one paragraph. All your work will have been in vain.
The advantage of having all white space (spaces, tabs, returns) ignored is that you can put your tags wherever you want. The following examples all produce the same output. Try them!

```
<h1>If music be the food of love, play on.</h1>
<h1>
If music be the food of love, play on.
</h1>
<h1>
If music be the food of love, play on. </h1>
<h1> If music be the food of love,
play on. </h1 >
```

Structuring Your HTML

The DOCTYPE Identifier

Although it's not a page structure tag, the XHTML 1.0 recommendation includes one additional requirement for your web pages. The first line of each page must include a DOCTYPE identifier that defines the XHTML 1.0 version to which your page conforms, and the document type definition (DTD) that defines the specification. This is followed by the <html>, <head>, and <body> tags. In the following example, the XHTML 1.0 Strict document type appears before the page structure tags:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/strict.dtd">
<html>
<head>
<title>Page Title</title>
</head>
<body>
...your page content...
</body>
</html>
```

Three types of HTML 4.01 document types are specified in the XHTML 1.0 specification:
Strict, Transitional, and Frameset.

The <html> Tag

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

The <head> Tag

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/transitional.dtd">
<html>
<head>
<title>This is the Title. It will be explained later on</title>
</head>
...your page...
</html>
```

The <body> Tag

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/transitional.dtd">
<html>
<head>
<title>This is the Title. It will be explained later on</title>
</head>
<body>
...your page...
</body>
</html>
```

The Title

Each HTML page needs a title to indicate what the page describes. It appears in the title bar of the browser when people view the web page.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/transitional.dtd">
<html>
<head>
<title>The Lion, The Witch, and the Wardrobe</title>
</head>
<body>
...your page...
</body>
</html>
```

Headings

Headings are used to add titles to sections of a page. HTML defines six levels of headings.

Heading tags look like the following:

```
<h1>Movies</h1>
<h2>Action/Adventure</h2>
<h3>Caper</h3>
<h3>Sports</h3>
<h3>Thriller</h3>
```

<h3>War</h3>
<h2>Comedy</h2>
<h3>Romantic Comedy</h3>
<h3>Slapstick</h3>
<h2>Drama</h2>
<h3>Buddy Movies</h3>
<h3>Mystery</h3>
<h3>Romance</h3>
<h2>Horror</h2>

Paragraphs

As of the HTML 4.01 standard, paragraph tags are two-sided (...), and indicates the beginning of the paragraph. The closing tag is no longer optional, so rather than using to indicate where one paragraph ends and another begins, you enclose each paragraph within a tag.

Input

The dragon fell to the ground, releasing an anguished cry and seething in pain. The thrust of Enigern's sword proved fatal as the dragon breathed its last breath. Now Enigern was free to release Lady Aelfleada from her imprisonment in the dragon's lair.

Image:

Images displayed on the Web should be converted to one of the formats supported by most browsers: GIF, JPEG, or PNG. GIF and JPEG are the popular standards, and every graphical browser supports them. PNG is a newer image format that was created in response to some patent issue s with the GIF format.

The most important attribute of the <img> tag is src, which is the URL of the image you want to include. Paths to images are derived in the same way as the paths in the href attribute of links. So, to point to a GIF file named image.gif in the same directory as the HTML document, you can use the following HTML tag:

Input:

<h1>Welcome to The Halloween House of Terror!!</h1>

Output:

Links:

To create a link in an HTML page, you use the HTML link tag <a>.... The <a> tag often is called an anchor tag because it also can be used to create anchors for links.

Input

Go back to
Main Menu

Lists:

HTML 4.01 defines these three types of lists:

• Numbered or ordered lists, which are typically labeled with numbers

• Bulleted or unordered lists, which are typically labeled with bullets or some other symbol

• Glossary lists, in which each item in the list has a term and a definition for that term, arranged so that the term is somehow highlighted or drawn out from the text

List Tags

All the list tags have the following common elements:

• The entire list is surrounded by the appropriate opening and closing tag for the type of list (for example, <ul> and for unordered lists, or <ol> and for ordered lists).
• Each list item within the list has its own tag:
<dt> and <dd> for the glossary lists, and <li> for all the other lists.

Input

• Installing Your New Operating System
•
• Insert the CD-ROM into your CD-ROM drive.
• Choose RUN.
• Enter the drive letter of your CD-ROM (example: D:\), followed by SETUP.EXE.
• Follow the prompts in the setup program.
• Reboot your computer after all files are installed.
• Cross your fingers.
•

Customizing Ordered Lists

You can customize ordered lists in two main ways: how they're numbered and the number with which the list starts. HTML 3.2 provides the type attribute that can take one of five values to de fine which type of numbering to use on the list:
• "1" Specifies that standard Arabic numerals should be used to number the list (that is, 1, 2, 3, 4, and so on)
• "a" Specifies that lowercase letters should be used to number the list (that is, a, b, c, d, and so on)
• "A" Specifies that uppercase letters should be used to number the list (that is, A, B, C, D, and so on)
• "i" Specifies that lowercase Roman numerals should be used to number the list (that is, i, ii, iii, iv, and so on)
• "I" Specifies that uppercase Roman numerals should be used to number the list (that is, I, II, III, IV, and so on)
You can specify types of numbering in the <ol> tag, as follows: <ol type="a">. By default type="1" is assumed.

Input

The Days of the Week in French:
<ol type="I">
Lundi
Mardi
Mercredi
Jeudi
Vendredi
Samedi
Dimanche

Input

The Last Six Months of the Year (and the Beginning of the NextYear):
<ol type="I" start="7">
July
August

September
October
November
December
<li type="1">January

Tables:

Table Parts

Before getting into the actual HTML code to create a table, let's look at the following terms so that we both know what we're talking about:

• The caption indicates what the table is about: for example, "Voting Statistics" or "Toy Distribution Per Room" Captions are optional.

• The table headings label the rows, columns, or both. Usually they're in an emphasized font that's different from the rest of the table. They're optional.

• Table cells are the individual squares in the table. A cell can contain normal table data or a table heading.

• Table data is the values in the table itself. The combination of the table headings and table data makes up the sum of the table.

The <table> Element

The to create a table in HTML, you use <table>...</table> element to enclose the code for an optional caption, and then add the contents of the table itself:

<table>
...table caption (optional) and contents...
</table>

Rows and Cells

The cells within each row are created using one of two elements:

• <th>...</th> elements are used for heading cells. Generally, browsers center the contents of a <th> cell and render any text in the cell in boldface.

• <td>...</td> elements are used for data cells. TD stands for table data.

Input

<tr>
<th>Name</th>
<td>Alison</td>
<td>Tom</td>
<td>Susan</td>
</tr>
<tr>
<th>Height</th>
<td>5'4"</td>
<td>6'0"</td>
<td>5'1"</td>
</tr>
<tr>
<th>Weight</th>
<td>140</td>
<td>165</td>

<td>97</td>
</tr>
<tr>
<th>Eye Color</th>
<td>Blue</td>
<td>Blue</td>
<td>Brown</td>
</tr>

Setting Table Widths

To make a table as wide as the browser window, you add the width attribute to the table, as shown in the following line of code:

<table border="1" width="100%">

Changing Table Borders

You can change the width of the border drawn around the table. If border has a numeric value, the border around the outside of the table is drawn with that pixel width. The default is border="1". border="0" suppresses the border, just as if you had omitted the border attribute altogether.

Input

<table border="10" width="100%">

Cell Padding

The cell padding attribute defines the amount of space between the edges of the cells and the content inside a cell.

Input

<table cellpadding="10" border="1">

Cell Spacing

Cell spacing is similar to cell padding except that it affects the amount of space between cells that is, the width of the space between the inner and outer lines that make up the table border.

Input

<table cellpadding="10" border="4" cellspacing="8">

Spanning Multiple Rows or Columns

The tables you've created up to this point all had one value per cell or the occasional empty cell. You also can create cells that span multiple rows or columns within the table. Those spanned cells then can hold headings that have subheadings in the next row or column or you can create other special effects within the table layout.

Input

<html>
<head>
<title>Row and Column Spans</title>
</head>
<body>
<table border="1" summary="span example">
<tr>
<th colspan="2">Gender</th>
</tr>
<tr>
<th>Male</th>
<th>Female</th>

</tr>
<tr>
<td>15</td>
<td>23</td>
</tr>
</table>
</body>
</html>

Forms:

Using the <form> Tag

To accept input from a user, you must wrap all of your input fields inside a <form> tag. The purpose of the <form> tag is to indicate where and how the user's input should be sent. First, let's look at how the <form> tag affects page layout. Forms are block-level elements.

Input

Please enter your username <form><input /> and password <input /></form> to log in.

The two most commonly used attributes of the <form> tag are action and method. Both of these attributes are optional. The following example shows how the <form> tag is typically used:

<form action="someaction" method="get or post">
content, form controls, and other HTML elements
</form>

action specifies the URL to which the form is submitted. Again, remember that for the form to be submitted successfully, the script must be in the exact location you specify and must work properly.

The method attribute supports two values: get or post. The method indicates how the form data should be packaged in the request that's sent back to the server. The get method appends the form data to the URL in the request.

Creating Text Controls

Text controls enable you to gather information from a user in small quantities. This control type creates a single-line text input field in which users can type information, such as their name or a search term.

Input

Enter your pet's name:
<input type="text" name="petname" /></ p>

Creating Password Controls

The password and text field types are identical in every way except that the data entered in a password field is masked so that someone looking over the shoulder of the person entering information can't see the value that was typed into the field.

Input

Enter your password: <input type="password" name="userpassword" size="8" maxlength="8" />

Creating Submit Buttons

Submit buttons are used to indicate that the user is finished filling out the form. Setting the type attribute of the form to submit places a submit button on the page with the

default label determined by the browser, usually Submit Query. To change the button text, use the value attribute and enter your own label, as follows:

<input type="submit" value="Send Form Data" />

Creating Reset Buttons

Reset buttons set all the form controls to their default values. These are the values included in the value attributes of each field in the form (or in the case of selectable fields, the values that are preselected). As with the Submit button, you can change the label of a Reset button to one of your own choosing by using the value attribute, like this:

<input type="reset" value="Clear Form" />

Creating Check Box Controls

Check boxes are fields that can be set to two states: on and off. To create a check box, set the input tag's type attribute to checkbox. The name attribute is also required, as shown in the following example:

Input

Check to receive SPAM email <input type="checkbox" name="spam" />

Creating Radio Buttons

Radio buttons, which generally appear in groups, are designed so that when one button in the group is selected, the other buttons in the group are automatically unselected. They enable you to provide users with a list of options from which only one option can be selected. To create a radio button, set the type attribute of an <input> tag to radio. To create a radio button group, set the name attributes of all the fields in the group to the same value. To cre ate a radio button group with three options, the following code is used:

Input

Select a color:
<input type="radio" name="color" value="red" /> Red
<input type="radio" name="color" value="blue" /> Blue
<input type="radio" name="color" value="green" /> Green

Creating Menus with select and option

The select element creates a menu that can be configured to enable users to select one or more options from a pull-down menu or a scrollable menu that shows several options at once. The <select> tag defines how the menu will be displayed and the name of the parameter associated with the field. The <option> tag is used to add selections to the menu. The default appearance of select lists is to display a pull-down list that enables the user to select one of the options. Here's an example of how one is created:

Input

Please pick a travel destination:
<select name="location">
<option>Indiana</option>
<option>Fuji</option>
<option>Timbuktu</option >
<option>Alaska</option>
</select>

Frames for designing a good website:

The first HTML document you need to create is called the frameset document. In this document, you define the layout of your frames, and the locations of the documents to be

initially loaded in each frame. Each of the three HTML documents other than the frameset document, the ones that load in the frames, contain normal HTML tags that define the contents of each separate frame area. These documents are referenced by the frameset document.

The <frameset> Tag

To create a frameset document, you begin with the <frameset> tag. When used in an HTML document, the <frameset> tag replaces the <body> tag, as shown in the following code:

```
<html>
<head>
<title>Page Title</title>
</head>
<frameset>
.. your frameset goes here ...
</frameset>
</html>
```

It's important that you understand up front how a frameset document differs from a normal HTML document. If you include a <frameset> tag in an HTML document, you cannot include a <body> tag also.

The cols Attribute

When you define a <frameset> tag, you must include one of two attributes as part of the tag definition. The first of these attributes is the cols attribute, which takes the following form:

```
<frameset cols="column width, column width, ...">
```

Input

```
<html>
<head>
<title>Three Columns</title>
</head>
<frameset cols="100,50%,*">
<frame src="leftcol.html">
<frame src="midcol.html">
<frame src="rightcol.html">
</frameset>
</html>
```

The rows Attribute

The rows attribute works the same as the cols attribute, except that it splits the screen into horizontal frames rather than vertical ones. To split the screen into two frames of equal height, you would write the following:

Input

```
<html>
<head>
<title>Two Rows</title>
</head>
<frameset rows="50%,50%">
<frame src="toprow.html">
<frame src="botrow.html">
</frameset>
```

</html>

The <frame> Tag

After you have your basic frameset laid out, you need to associate an HTML document with each frame by using the <frame> tag, which takes the following form:

<frame src="document URL">

For each frame defined in the <frameset> tag, you must include a corresponding <frame> tag, as shown in the following:

Input

```
<html>
<head>
<title>The FRAME Tag</title>
</head>
<frameset rows="*,*,*">
<frame src="document1.html" />
<frame src="document2.html" />
<frame src="document3.html" />
</frameset>
</html>
```

Changing Frame Borders

Start with the <frame> tag. By using two attributes, bordercolor and frameborder, you can turn borders on and off and specify their color. You can assign bordercolor any valid color value, either as a name or a hexadecimal triplet. frameborder takes two possible values:

1(to display borders) or 0 (to turn off the display of borders).

```
<html>
<head>
<title>Conflicting Borders</title>
</head>
<frameset frameborder="0" rows="*,*,*">
<frame frameborder="1" bordercolor="yellow" src="document1.html">
<frame bordercolor="#cc3333" src="document2.html">
<frame src="document3.html">
</frameset>
</html>
```

long questions:

1. What is internet? Want is WWW? What is the difference between them?
2. What are the different lists available explain briefly.
3. Explain the different tags and attributes available in table briefly.
4. What are the different tags available to create the elements of a form explain in detail.

Java Script, CSS and DOM

Java Script:

Programming Fundamentals:

JavaScript, originally called LiveScript, was developed by Brendan Eich at Netscape in 1995 and was shipped with Netscape Navigator 2.0 beta releases. JavaScript programs are used to detect and react to user-initiated events, such as a mouse going over a link or graphic. They can improve a Web site with navigational aids, scrolling messages and rollovers, dialog boxes, dynamic images, shopping carts, and so forth.

Client-side JavaScript programs are embedded in an HTML document between HTML head tags <head> and </head> or between the body tags <body> and </body>. Many developers prefer to put JavaScript code within the <head> tags, and at times, as you will see later, it is the best place to store function definitions and objects. If you want text displayed at a specific spot in the document, you may want to place the JavaScript code within the <body> tags. Or you may have multiple scripts within a page, and place the JavaScript code within both the <head> and <body> tags. In either case, a JavaScript program starts with a <script> tag, and and ends with a </script> tag. And if the JavaScript code is going to be long and involved, or may be reused, it can be placed in an external file (ending in .js) and loaded into the page.

```
1 <html>
2 <head><title>First JavaScript Sample</title></head>
3 <body bgcolor="yellow" text="blue">
4 <script language = "JavaScript" type="text/javascript">
4 document.writeln("<h2>Welcome to the JavaScript
World!</h1>");
5 </script>
6 <font size="+2">This is just plain old HTML stuff.</font>
7 </body>
8 </html>
```

Statements and Expressions:

Comments

Single line comments start with a double slash:

```
// This is a comment
```

For a block of comments, use the /* */ symbols:

```
/* This is a block of comments
that continues for a number of lines
*/
```

The <script> Tag

<script>
JavaScript statements...
</script>

<script>
document.write("Hello, world!
");
</script>

Attributes

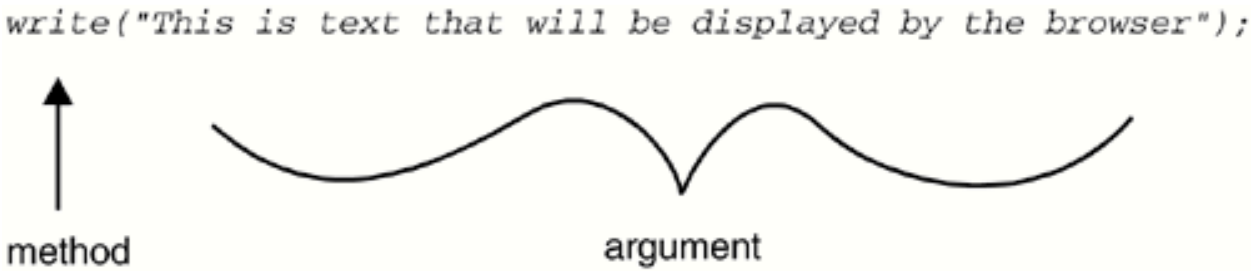
The <script> tag also has attributes to modify the behavior of the tag. The attributes are

- language
- type
- src

<script language="JavaScript"
type="text/javascript"
src="directory/sample.js">
</script>

String Concatenation

Concatenation is caused when two strings are joined together. The plus (+) sign is used to concatenate strings; for example:
"hot" + "dog or "San Francisco" + "</br>"
The write() and writeln() Methods



<html>
<head><title>Printing Output</title></head>
<body bgcolor="yellow" text="blue">
Comparing the document.write and document.writeln
 methods
<script language="JavaScript">
document.write("<h3>One, "); // No newline
document.writeln("Two, ");
document.writeln("Three, ");
document.write("Blast off....
"); // break tag
document.write("The browser you are using is " +
navigator.userAgent + "
");

```
</script>
<pre>
<script language="JavaScript">
document.writeln("With the <em>HTML &lt;pre&gt;
</em> tags, ");
document.writeln("the <em>writeln</em> method produces a newline.");
document.writeln("Slam");
document.writeln("Bang");
document.writeln("Dunk!");
</script>
</pre>
</body></html>
```

Data Types

Primitive Data Types

Primitive data types are the simplest building blocks of a program. They are types that can be assigned a single literal value such as the number 5.7, or a string of characters such as "hello". JavaScript supports three core or basic data types:

- numeric
- string
- Boolean

In addition to the three core data types, there are two other special types that consist of a single value:

- null
- undefined

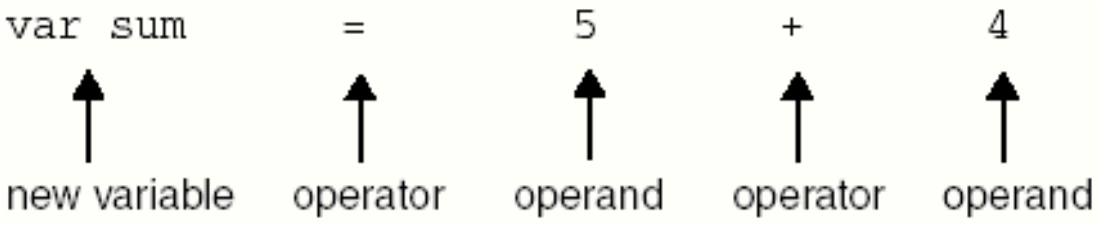
Variables

Variables are fundamental to all programming languages. They are data items that represent a memory storage location in the computer. Variables are containers that hold data such as numbers and strings. Variables have a name, a type, and a value. JavaScript variables can be assigned three types of data:

- numeric
- string
- Boolean

Operators:

Data objects can be manipulated in a number of ways by the large number of operators provided by JavaScript. Operators are symbols, such as +, –, =, >, and <, that produce a result based on some rules.



Precedence and associativity

Operator Description Associativity

() Parentheses Left to right

++ — Auto increment, decrement Right to left

! Logical NOT Right to left

* / % Multiply, divide, modulus Left to right

+ – Add, subtract Left to right

+ Concatenation Left to right

<<= Less than, less than equal to Left to right

>>= Greater than, greater than equal to Left to right

= = != Equal to, not equal to Left to right

= = = != = Identical to (same type), not identical to Left to right

&Bitwise AND Left to right

Operator Description Associativity

Bitwise OR

^ Bitwise XOR

~ Bitwise NOT

<< Bitwise left shift

>> Bitwise right shift

>>> Bitwise zero-filled, right shift

&& Logical AND Left to right

Logical OR Left to right

? : Ternary, conditional Right to left

= += – = *= /= %= <<= >>= Assignment Right to left

, (comma)

Types of Operators:

Arithmetic Operators

Arithmetic operators.

Operator/Operands	Function
x + y	Addition
x – y	Subtraction
x * y	Multiplication
x / y	Division
x % y	Modulus

Shortcut Assignment Operators

Assignment operators.

Operator Example Meaning

= var x = 5; Assign 5 to variable x.

+= x += 3; Add 3 to x and assign result to x.

–= x –= 2; Subtract 2 from x and assign result to x.

*= x *= 4; Multiply x by 4 and assign result to x.

/= x /= 2; Divide x by 2 and assign result to x.

**= x **= 2; Square x and assign result to x.

%= x %= 2 Divide x by 2 and assign remainder to x.

Auto increment and Auto decrement Operators

To make programs easier to read, to simplify typing, and, at the machine level, to produce more efficient code, the auto increment (++) and auto decrement (– –) operators are provided. Auto increment and auto decrement operators.

Operator Function What It Does Example

++x Pre-increment Adds 1 to x x = 3; x++; x is now 4

x++ Post-increment Adds 1 to x x = 3; ++x; x is now 4

– –x Pre-decrement Subtracts 1 from x x = 3; x – –; x is now 2

x– – Post-decrement Subtracts 1 from x x = 3; – –x; x is now 2