Fundamentals of Web Programming

DCAP202

Editor
Anuj Sharma

LOVELY
PROFESSIONAL
UNIVERSITY

LOVELY PROFESSIONAL UNIVERSITY

FUNDAMENTALS OF WEB PROGRAMMING

Edited By
Anuj Sharma

ISBN: 978-93-87034-78-5

Printed by
EXCEL BOOKS PRIVATE LIMITED
Regd. Office: E-77, South Ext. Part-I, Delhi-110049
Corporate Office: 1E/14, Jhandewalan Extension, New Delhi-110055
+91-8800697053, +91-011-47520129
info@excelbooks.com/projects@excelbooks.com
internationalalliance@excelbooks.com
www.excelbooks.com

for
Lovely Professional University
Phagwara

CONTENTS

SYLLABUS

Fundamentals of Web Programming

Objectives: To impart the skills needed for web programming, web administration, and web site development.

Sr. No.	Topics
1.	**Internet Basic:** Basic Concepts, Communicating on the Internet, Internet Domains, Establishing connectivity to the Internet, Client IP Address, IP Address, TCP/IP
2.	**HTML:** Introduction, Web Server, Web Client/ Browser, HTML Tags
3.	**HTML:** Commonly used HTML Commands, Structure of HTML Program, Formatting, Text Styles, Text Effects
4.	**HTML:** HTML Lists, Types of lists, Adding graphics to HTML Document
5.	**HTML:** Creating tables, Linking documents, Frames
6.	**Introduction to Javascript:** Javascript and Web. <script> Tag and browsers compatibility. Data types: Numeric, Text, Boolean, Type casting, Arrays
7.	**Javascript:** Operators and expressions in Javascript, Programming Constructs, Conditional and Looping Statements, Functions, User defined functions, Dialog Boxes
8.	**DOM Model:** Understanding DOM Model, Objects in HTML, Browser Objects, window, history, location, navigator, document object. Handling Events using Javascript
9.	**HTML Forms:** Properties and Methods, Button, Text, Text Area, Checkboxes, radio buttons, select and option elements, Built-In objects in Javascript, String Object, Math Object, Date Object, User Defined objects
10.	**DHTML:** Cascading Style Sheets, Class, External Style Sheets

Unit 1: Internet Basic

Notes

CONTENTS

Objectives

After studying this unit, you will be able to:

- Discuss the basic concept of Internet
- Explain the communication on the Internet
- Discuss Internet Domain
- Describe establishing connectivity to the Internet

Introduction

The Internet carries a vast array of information resources and services, most notably the inter-linked hypertext documents of the World Wide Web (WWW) and the infrastructure to support electronic mail, in addition to popular services such as online chat, file transfer and file sharing, online gaming, and Voice over Internet Protocol (VoIP) person-to-person communication via voice and video.

1.1 Basic Concept

It is a network of networks that consists of millions of private and public, academic, business, and government networks of local to global scope that are linked by copper wires, fiber-optic cables, wireless connections, and other technologies.

1.1.1 Browsers

A browser is a software application which enables a user to display and interact with text, images, videos, music, and other information that could be on a website. Text and images on a web page can contain hyperlinks to other web pages at the same or different website.

Browsers allow a user to quickly and easily access information provided on many web pages at many websites by traversing these links.

1.1.2 Websites

A website is a collection of web pages (documents that are accessed through the Internet).

A web page is what you see on the screen when you type in a web address, click on a link, or put a query in a search engine. A web page can contain any type of information, and can include text, color, graphics, animation and sound.

When someone gives you their web address, it generally takes you to their website's home page, which should introduce you to what that site offers in terms of information or other services. From the home page, you can click on links to reach other sections of the site.

Caution A website can consist of one page, or of tens of thousands of pages, depending on what the site owner is trying to accomplish.

1.1.3 Web Addresses

A Web address, or domain name, is an address where you can be found online. It's how you'll express yourself through email or your website and it's what customers think of when trying to find you.

Self Assessment

Fill in the blanks:

1. A .... is a software application which enables a user to display and interact with text, images, videos, music, and other information that could be on a website.
2. A .... is what you see on the screen when you type in a web address, click on a link, or put a query in a search engine.
3. A ...., or domain name, is an address where you can be found online.

1.2 Communicating on Internet

Communication is the most popular use of the Internet, with email topping the list of all the technologies used. Some of the types of communication technologies used also include email discussion groups, Usenet news, chat groups, and IRC. These are unique to networked computer environments and have come into wide popularity because of the Internet. Other technologies, including video and audio conferencing and Internet telephony, are also available on the Internet. They require more multimedia capabilities of computer systems and are more taxing of network resources than the others. They also are adaptations of other technologies to the Internet.

Most of the technologies that are unique to the Internet require communication to be done in text—letters with some symbols and punctuation.

Notes Communicating effectively involves taking the time, except in informal communications, to use correct grammar, spelling, and punctuation and writing an appropriate message. When replying to a message include the pertinent parts of the message and use an appropriate and interesting subject header in any case.

Self Assessment

State True or False:

4. Communication technologies are unique to networked computer environments and have come into wide popularity because of the Internet.
5. Communication technologies does not require more multimedia capabilities of computer systems and are more taxing of network resources than the others.

1.3 Internet Domain

An internet domain name is an identification label that defines a realm of administrative autonomy, authority, or control in the Internet, based on the Domain Name System (DNS).

1.3.1 Domain Names

A domain name is an identification string that defines a realm of administrative autonomy, authority, or control on the Internet. Domain names are formed by the rules and procedures of the Domain Name System (DNS). Technically, any name registered in the DNS is a domain name.

Example: The domain name microsoft.com represents about a dozen IP addresses. Domain names are used in URLs to identify particular Web pages. For example, in the URL http://www.abc.com/index.html, the domain name is abc.com. Every domain name has a suffix that indicates which Top Level Domain (TLD) it belongs to. There are only a limited number of such domains. For example:

Figure 1.1: Examples of Domain Names

Gov	Government Agencies
Edu	Educational Institutions
Org	Organizational (nonprofit)
Mil	Military
Com	Commercial Business
Net	Network Organizations
ca	Canada
th	Thailand

1.3.2 Host Name

A name is a label that is used to distinguish one thing from another. A person's name, for instance, comprises a set of alphabetic characters that allows a person to be individually addressed. Computers are also named to differentiate one machine from another and to allow for such activities as network communication.

A hostname is a label that is assigned to a device connected to a computer network and that is used to identify the device in various forms of electronic communication such as the World Wide Web, e-mail or Usenet. Hostnames may be simple names consisting of a single word or phrase, or they may have appended a domain name, which is a name in a Domain Name System (DNS), separated from the host specific label by a period (dot). In the latter form, the hostname is also called a domain name.

Did u know? If the domain name is completely specified including a top-level domain of the Internet, then the hostname is said to be a Fully Qualified Domain Name (FQDN).

The true "name" a computer needs in order to communicate on a network is actually a set of numbers. The original computers connected as the Internet used small integers as the host number. For TCP/IP, the main protocol used by the Internet, each computer has a network IP address that follows a specific set of rules to assure its uniqueness and validity. (Additionally, port numbers further specify the access points for particular services on a computer).

1.3.3 Domain Name System

The DNS translates Internet domain and host names to IP addresses. DNS automatically converts the names we type in our Web browser address bar to the IP addresses of Web servers hosting those sites.

An often-used analogy to explain the Domain Name System is that it serves as the phone book for the Internet by translating human-friendly computer hostnames into IP addresses.

Example: The domain name www.example.com translates to the addresses 192.0.43.10 (IPv4) and 2001:500:88:200::10 (IPv6).

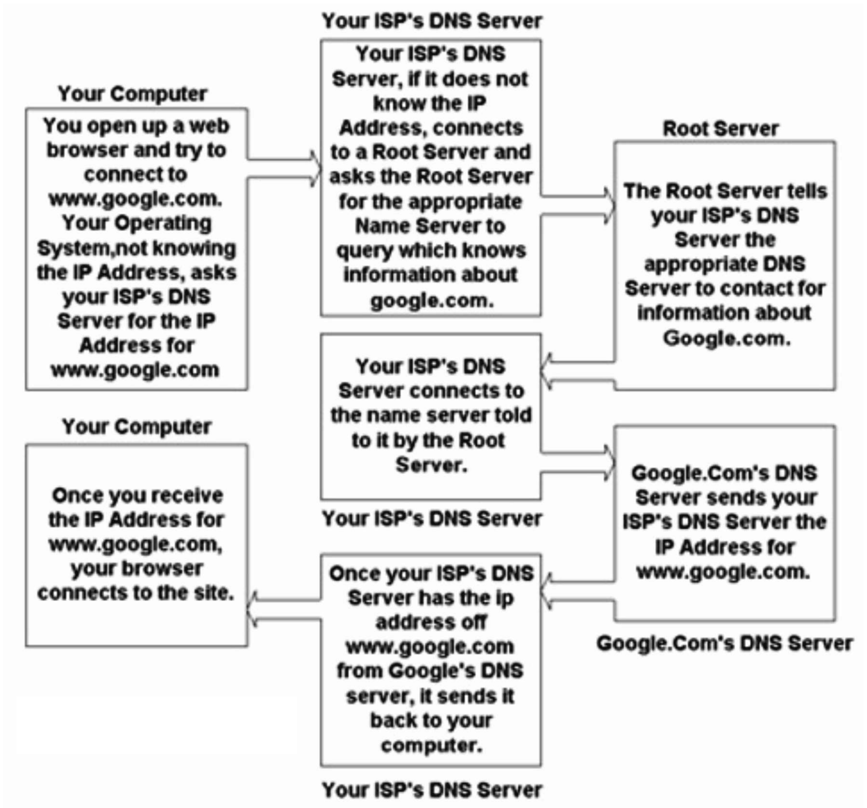
Notes

Unlike a phone book, the DNS can be quickly updated, allowing a service's location on the network to change without affecting the end users, who continue to use the same host name. Users take advantage of this when they use meaningful Uniform Resource Locators (URLs) and e-mail addresses without having to know how the computer actually locates the services.

Real Life Example

A lot of what has been discussed may be a bit confusing, so lets do a real life example. In the flowchart below labeled Figure 1.2, you will see a computer trying to connect to www.google.com and the steps it takes.

Figure 1.2: Computer Trying to Connect to www.google.com


Your Computer
You open up a web browser and try to connect to www.google.com. Your Operating System,not knowing the IP Address, asks your ISP's DNS Server for the IP Address for www.google.com
Your ISP's DNS Server
Your ISP's DNS Server, if it does not know the IP Address, connects to a Root Server and asks the Root Server for the appropriate Name Server to query which knows information about google.com.
Root Server
The Root Server tells your ISP's DNS Server the appropriate DNS Server to contact for information about Google.com.
Your ISP's DNS Server connects to the name server told to it by the Root Server.
Your ISP's DNS Server
Google.Com's DNS Server sends your ISP's DNS Server the IP Address for www.google.com.
Google.Com's DNS Server
Once your ISP's DNS Server has the ip address off www.google.com from Google's DNS server, it sends it back to your computer.
Your ISP's DNS Server
Your Computer
Once you receive the IP Address for www.google.com, your browser connects to the site.

We will discuss these steps below:

1. A User opens a web browser and tries to connect to www.google.com. The operating system not knowing the IP Address for www.google.com, asks the ISP's DNS Server for this information.
2. The ISP's DNS Server does not know this information, so it connects to a Root Server to find out what name server, running somewhere in the world, know the information about google.com.
3. The Root Server tells the ISP's DNS Server to contact a particular name server that knows the information about google.com.
4. The ISP's DNS Server connects to Google's DNS server and asks for the IP Address for www.google.com.
5. Google's DNS Server responds to the ISP's DNS server with the appropriate IP Address.

6. The ISP's DNS Server tells the User's operating system the IP Address for google.com.
7. The operating system tells the Web Browser the IP Address for www.google.com.
8. The web browser connects and starts communication with www.google.com.

Self Assessment

Fill in the blanks:

6. A .... is an identification string that defines a realm of administrative autonomy, authority, or control on the Internet.
7. A .... is a label that is assigned to a device connected to a computer network and that is used to identify the device in various forms of electronic communication.

1.4 Establishing Connectivity to the Internet

All modern computers and laptops are capable of connecting to the internet, as are many other devices, including mobiles, tablets, e-readers, televisions, video games consoles.

There are two ways of getting the internet at home. The most popular way is to have your telephone line (also known as a 'landline') converted to broadband so that it can carry normal phone calls and internet data at the same time.

However, if you don't have a landline or if you want to be able to use the internet when you're out and about, you might prefer mobile internet from one of the mobile network providers. This can be used anywhere there's a mobile signal but does tend to be slower and more expensive than broadband through a landline.

Follow these step-by-step instructions to connect to the internet

Step 1: Choose an Internet Service Provider (ISP). This could be the company that provides your telephone line or it could be one of the many independent providers. To help you choose, have a look at one of the many comparison websites and ask people you know for their opinion.

Some ISPs, particularly the mobile suppliers, impose a monthly download limit or usage allowance. This is measured in gigabytes (Gb). To put this in context, 1Gb will allow you to visit approximately 10,500 webpages or download around 205 songs.

Step 2: Having chosen an ISP and signed the contract, you'll have to wait a few days while your line is converted to broadband. During this time, you should receive a letter with your username and password and the hardware you'll need: a small box called a 'router' and its attachments.

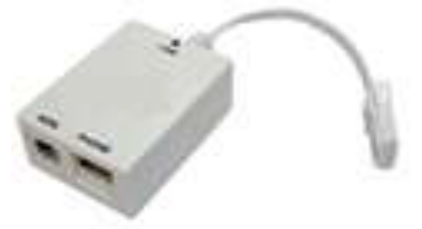
Step 3: Once you're told that your broadband is active, you can set up your router. It should have come with three cables:

- a network cable to connect the router to your computer
- a power cable
- a cable that will go between your router and a microfilter (see below).
- Plug one end of the network cable into the appropriately shaped socket in the router, and the other end in a similarly appropriately shaped socket in your computer.

Take the power cable and plug one end in the router and the other in a nearby power point.

Notes

Figure 1.3: Microfilter



You should have also received a microfilter. This splits the signal in the telephone wire in two: voice and broadband. You plug the dangly end of the microfilter into your telephone socket. Then in the sockets at the other end, you plug in (1) the cable from your own telephone and (2) the cable that came with your router. As these two sockets are different shapes, you can't plug a cable into the wrong socket. Finally, plug in the other end of the router cable into the router itself.

You'll also need to install a microfilter in any other telephone socket in the house that's in use. Not doing this can result in loss of internet speed and interference on the line.

Caution If no router is being used with a socket, you'll leave one of the microfilter's sockets empty.

Step 4: When you get the router, you should also receive a CD. Once you've set up the router, all you need to do is put the CD into your computer and follow the step-by-step instructions. If you don't want to do this yourself, some companies offer a home installation service and, for an extra cost, will send an engineer to set up your broadband connection for you.

1.4.1 Enable and Disable Automatic Wireless Network Configuration

To enable automatic configuration, ensure the "Use Windows to configure my wireless network settings" checkbox is checked on the Wireless Networks tab of the Wireless Network Connection properties window. Automatic wireless Internet/Wi-Fi network configuration will be disabled if this checkbox is unchecked. You must be logged on with Windows XP administrative privileges to enable/disable this feature.

1.4.2 Available Networks

The Wireless Networks tab allows you to access the set of "Available" networks. Available networks represent those active networks currently detected by Windows XP. Some Wi-Fi networks may be active and in range but not appear under Available networks. This occurs when a wireless router or access point has SSID broadcast disabled.

Whenever your network adapter detects newly available Wi-Fi networks, you will see an alert in the lower-right corner of the screen allowing you to take action if necessary.

1.4.3 Preferred Network

In the Wireless Networks tab, you can build a set of so-called "Preferred" networks when automatic wireless configuration is active. This list represents a set of known Wi-Fi routers or access points you wish to automatically connect to in future. You can "Add" new networks to this list by specifying the network name (SSID) and appropriate security settings of each.

The order Preferred networks determines the order that Windows XP will automatically attempt when seeking to make a wireless/Internet connection. You can set this order to your preference, with the limitation that all infrastructure mode networks must appear ahead of all ad hoc mode networks in the Preferred list.

Self Assessment

Fill in the blanks:

8. .... splits the signal in the telephone wire in two: voice and broadband.
9. The .... tab allows you to access the set of "Available" networks.

1.5 Client IP Address

It's pretty common the need for a web application to be able to detect the IP address of a client. The IP address could be used either for statistic or authentication purposes.

You can also use the IP address to block multiple logins with the same credentials. This could be useful for sites that offer content to users that have paid a subscription.

1.5.1 Using the Code

The code to do so is really easy, all you need to do is check a bunch of server variables. The good thing is that those variables are not platform specific, so an equivalent code could also work with PHP.

```
Function ClientIpAddress(ByVal myRequest As HttpRequest)
       Dim myIP As String = ""
       If (myIP = "") Then
       myIP = myRequest.ServerVariables("HTTP_CLIENT_IP")
       If (myIP = "") Then myIP =
       myRequest.ServerVariables("HTTP_X_FORWARDED_FOR")
       If (myIP = "") Then myIP =
       myRequest.ServerVariables("HTTP_X_FORWARDED")
       If (myIP = "") Then myIP =
       myRequest.ServerVariables("HTTP_ X_CLUSTER_CLIENT_IP")
       If (myIP = "") Then
       myIP = myRequest.ServerVariables("HTTP_FORWARDED_FOR")
       If (myIP = "") Then
       myIP = myRequest.ServerVariables("HTTP_FORWARDED")
       If (myIP = "") Then
       myIP = myRequest.ServerVariables("REMOTE_ADDR")
 Return myIP
End Function
```

Sometimes though we do not have the actual IP address but we have the host name of the ISP provider of the client. The host name can be easily resolved into an IP address easily. Using the System.Net.Dns.GetHostEntry function we get a list of all the IP addresses this hostname resolves to.

```
Function Host2Ip(ByVal HostName As String) As String
 Try
 Dim myIPs As System.Net.IPHostEntry
 myIPs = System.Net.Dns.GetHostEntry(HostName)
```

```
  Host2Ip  =  myIPs.AddressList(0).ToString()
  Catch  ex  As  Exception
  'Handle  the  error
  End  Try
End  Function
```

Finally, although this approach works much better than just looking at REMOTE_ADDR, still it's far from a full proof solution, simply because it relies on the HTTP header information which can be easily manipulated.

Self Assessment

Fill in the blanks:

10. The IP address could be used either for .... or authentication purposes.
11. Using the .... function we get a list of all the IP addresses the hostname resolves to.

1.6 IP Address

An IP address is an identifier for a computer or device on a TCP/IP network. Networks using the TCP/IP protocol route messages based on the IP address of the destination.

An **Internet Protocol address** (**IP address**) is a numerical label assigned to each device (e.g., computer, printer) participating in a computer network that uses the Internet Protocol for communication. An IP address serves two principal functions: host or network interface identification and location addressing. Its role has been characterized as follows: "*A name indicates what we seek. An address indicates where it is. A route indicates how to get there.*"

1.6.1 Network Part of IP Address

The format of an IP address is a 32-bit numeric address written as four numbers separated by periods. Each number can be zero to 255.

Example: 1.160.10.240 could be an IP address.

Within an isolated network, you can assign IP addresses at random as long as each one is unique. However, connecting a private network to the Internet requires using registered IP addresses (called Internet addresses) to avoid duplicates.

1.6.2 Local Host Part of the IP Address

A localhost is, put plainly, the standard hostname given to the address assigned to the loopback network interface. Translated into an IP address, a localhost is always designated as 127.0.0.1. This may be overwhelming, so let's look at what these things mean.

Hostname is the term for the label that is assigned to any device connected to a computer network and is used to identify the device across the web. A hostname usually consists of a single word or phrase and can be followed by a Domain Name System domain at the end. That is, a website. Used this way, the hostname is called a domain name.

Example: An example of using a hostname this way would be: hostname.domain.

Notes

1.6.3 IP Address Classes and their Formats

IP addresses are 32 bit integers which are represented in the familiar dot based notation. The dot based notation is nothing but a decimal representation for each byte of the IP address.

Example: An IP address with a hex value of 0x800A080B is represented as 128.10.8.11.

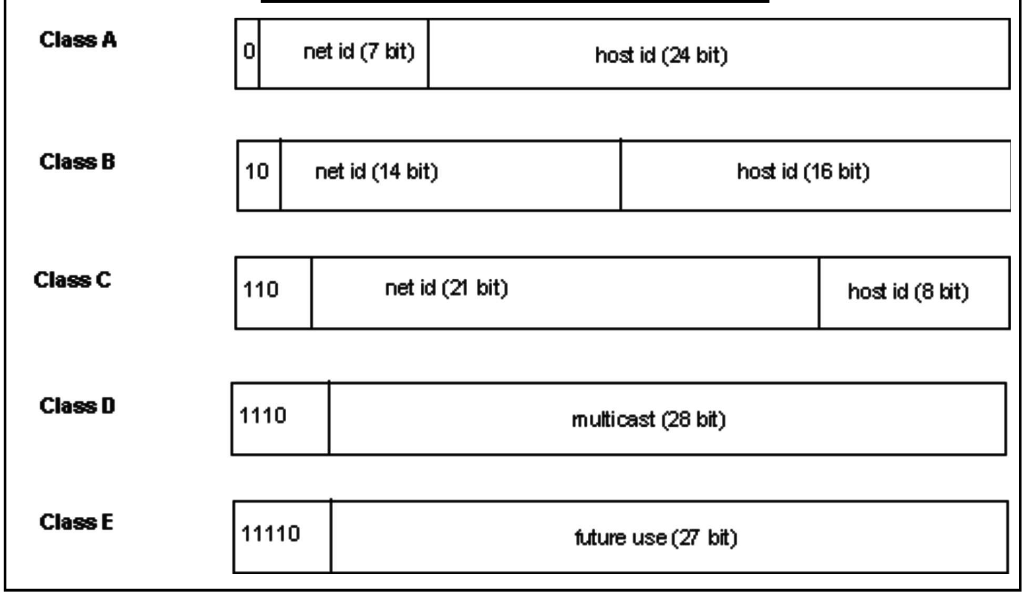
The internet, as the name suggests, is a network of networks. Thus to uniquely identify a host on the internet, one needs to know the network's id and the host's id in the network. Thus IP address consist of two components, the network id and the host id.

Did u know? The network id is the number assigned to a network in the internet. Host id represents the id assigned to a host in the network.

Figure 1.4 shows different classes of IP addresses. These addresses differ in the number of bits assigned to the network and host ids. Different classes of addresses serve different needs.

Example: A class A IP address is suitable when the internet consists of a small number of networks but each network consists of a large number of hosts. On the other extreme, class C addressing is suitable for internets with a very large number of networks, with a small number of hosts per network.

Figure 1.4: Different Classes of IP Addresses



1.6.4 Relation of IP Address to Physical Address

Network device has two types of addresses, one called the logical address — in most cases this is the IP address — and the other one being the physical address — also known as the MAC address.

The IP address is an address bound to the network device, i.e., computer, via software. In a Windows-powered computer, the Windows operating system allows the user to configure the IP address the specific workstation will have. This IP address is used to allow all network aware programs, i.e., Internet Explorer, Netscape, Outlook, etc. to use this address when communicating with other hosts. The seventh layer in the OSI model has the IP addresses.

The MAC address is a hardware address, which means it is unique to the network card installed on your PC. No two devices on a local network should ever have the same MAC address. In the unlikely event this occurs, the two devices will have major communication problems. During the manufacturing process, the vendor "burns" a specific MAC address into each network card's ROM. When the serial numbers have all been used, they start from the beginning, as it's very unlikely anyone would buy two network cards from the same vendor, and they will contain the same MAC address.

Notes You should remember that a IP address is a logical address which is configured via the operating system, while the MAC address is a hardware address, burnt into the network card's ROM during the manufacturing process.

1.6.5 Static Vs Dynamic IP Address

An IP address can be static or dynamic.

A *static IP address* will never change and it is a permanent Internet address.

A *dynamic IP address* is a temporary address that is assigned each time a computer or device accesses the Internet.

Task Compare static and dynamic IP address.

Self Assessment

Fill in the blanks:

12. An .... is an identifier for a computer or device on a TCP/IP network.
13. A .... is the standard hostname given to the address assigned to the loopback network interface.

1.7 TCP/IP (Transmission Control Protocol/Internet Protocol)

TCP/IP is made up of two acronyms, TCP, for Transmission Control Protocol, and IP, for Internet Protocol. TCP handles packet flow between systems and IP handles the routing of packets.

All modern networks are now designed using a layered approach. Each layer presents a predefined interface to the layer above it. By doing so, a modular design can be developed so as to minimize problems in the development of new applications or in adding new interfaces.

1.7.1 TCP/IP Protocol Architecture

The ISO/OSI protocol with seven layers is the usual reference model. Since TCP/IP was designed before the ISO model was developed it has four layers; however the differences between the two are mostly minor. Below, is a comparison of the TCP/IP and OSI protocol stacks:

```
OSI Protocol Stack
```

Notes

```
7. Application — End user services such as email.
6. Presentation — Data problems and data compression
5. Session — Authenication and authorization
4. Transport — Guarantee end-to-end delivery of packets
3. Network — Packet routing
2. Data Link — Transmit and receive packets
1. Physical — The cable or physical connection itself.
TCP/IP Protocol Stack.
5. Application — Authenication, compression, and end user services.
4. Transport — Handles the flow of data between systems and
provides access to the network for applications via
the (BSD socket library)
3. Network — Packet routing
2. Link — Kernel OS/device driver interface to the network
interface on the computer.
```

Task Make a report on the layers of TCP/IP Protocol Architecture.

Self Assessment

Fill in the blanks:

14. TCP handles .... between systems and IP handles the routing of packets.
15. The .... protocol with seven layers is the usual reference model.

1.8 Summary

- The Internet carries a vast array of information resources and services.
- A Web address, or domain name, is an address where you can be found online.
- A **hostname** is a label that is assigned to a device connected to a computer network and that is used to identify the device.
- The **DNS** translates Internet domain and host names to IP addresses.
- The order Preferred networks determines the order that Windows XP will automatically attempt when seeking to make a wireless/Internet connection.
- An IP address is an identifier for a computer or device on a TCP/IP network. Networks using the TCP/IP protocol route messages based on the IP address of the destination.
- IP addresses are 32 bit integers which are represented in the familiar dot based notation.
- Network device has two types of addresses, one called the logical address — in most cases this is the IP address — and the other one being the physical address — also known as the MAC address.

1.9 Keywords

Browsers: A browser is a software application which enables a user to display and interact with text, images, videos, music, and other information that could be on a website.

Domain name: A domain name is an identification string that defines a realm of administrative autonomy, authority, or control on the Internet.

Dynamic IP address: It is a temporary address that is assigned each time a computer or device accesses the Internet.

*H**ostname:*** It is a label that is assigned to a device connected to a computer network and that is used to identify the device in various forms of electronic communication such as the World Wide Web

Static IP: This address will never change and it is a permanent Internet address.

Web Addresses: A Web address, or domain name, is an address where you can be found online.

Web page: A web page is what you see on the screen when you type in a web address, click on a link, or put a query in a search engine.

Website: A website is a collection of web pages (documents that are accessed through the Internet).

1.10 Review Questions

1. Discuss www and Internet.
2. How to communicate on the Internet?
3. Explain the Domain Name system.
4. Write the procedure of connectivity to the Internet.
5. Write a note on IP addressing.
6. Describe the classes of IP address.
7. Write a note on TCP/IP.
8. Make distinction between website and webpage.
9. Explain the concept of domain name system with example.
10. Domain names are formed by the rules and procedures of the Domain Name System (DNS). Comment.

Answers: Self Assessment

1.	browser	2.	web page
3.	Web address	4.	True
5.	False	6.	domain name
7.	hostname	8.	Microfilter
9.	Wireless Networks	10.	Statistic
11.	System.Net.Dns.GetHostEntry	12.	IP address

Notes

13. localhost
14. packet flow
15. ISO/OSI

1.11 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://domainnamesissue.blogspot.in

http://ictmanual.net

http://openbookproject.net/

http://searchnetworking.techtarget.com

http://www.freeservers.com/

http://www.webliminal.com/

http://www.webopedia.com/

Unit 2: HTML Introduction

CONTENTS

Objectives

After studying this unit, you will be able to:

- Define markup languages and Web server
- Describe the Web browser
- Discuss HTML tags

Introduction

When you look at a web page in a web browser, you see, at the simplest level, words. These words usually have some style characteristics, such as different font sizes and colors. In many cases a page also displays images or maybe video. Sometimes there is a form where you can enter (or search) for information, or customize the display of the page to your liking. Often a page contains animated content and content that changes while the rest of the page remains the same.

Several technologies (such as CSS, JavaScript, Flash, AJAX, JSON) can be used to define the elements of a web page. However, at the very lowest level, a web page is defined using **HTML** (**HyperText Markup Language)**. Without HTML, there is no web page.

HTML is a **markup language**. It tells the web browser how to display content. HTML separates "content" (words, images, audio, video, and so on) from "presentation" (the definition of the type of content and the instructions for how that type of content should be displayed). HTML uses a predefined set of elements to identify content types. Elements contain one or more "tags" that contain or express content. Tags are surrounded by angle brackets, and the "closing" tag (the one that indicates the end of the content) is prefixed by a forward slash.

2.1 Basic of Markup Languages

A markup language is a language that annotates text so that the computer can manipulate the text. Most markup languages are human readable because the annotations are written in a way to distinguish them from the text.

Example: With HTML, XML, and XHTML, the markup tags are < and >.

Any text that appears within one of those characters is considered part of the markup language and not part of the annotated text.

Example:

<p>this is a paragraph of text written in HTML</p>

When you format text to be printed (or displayed on a computer or TV), you need to distinguish between the text itself and the instructions for printing the text.

Did u know? The markup is the instructions for displaying or printing the text.

Markup doesn't have to be computer readable. Annotations done in print or in a book are also markup.

Example: Many students in school will highlight certain phrases in their text books. This indicates that the highlighted text is more important than the surrounding text. The highlight color is markup.

Markup becomes a language when rules are codified around how to write and use the markup. That same student could have their own "note taking markup language" if they codified rules like "purple highlighter is for definitions, yellow highlighter is for exam details, and pencil notes in the margins are for additional resources." But most markup languages are defined by an outside authority for use by many different people.

2.1.1 Types of Markup Languages

Most of the content we see on the web exists because markup languages like HTML and XML exist. Well at least that's what we are thought about in academics. That we know, but some of you might not know that this class of languages in not limited to two or three. Lets look into a few more that are used extensively:

DHTML: Dynamic Hyper Text Markup Language can be described as a combination of several technologies like HTML client-side java script and cascading Style Sheets. Most of the pages and multimedia content on the web are created using DHTML.

XHTML: eXtensible Hyper Text Markup Language. Traditional HTML does not impost much structural strictness, sometimes resulting in poorly-displayed pages. The use of XHTML enables content to be displayed similarly across different browsers.

VoiceXML: Used in Voice interaction between humans and computer, mainly in systems that enable you to, for example check your credit card balance over the phone.

Caution The dialogue management and speech recognition- is defined by voiceXML

LaTeX: A document markup language used mainly bye mathematicians, authors, etc. to typeset their content. It is suitable for representing mathematical formulas.

Task Make a report on the different types of HTML with their examples.

2.1.2 Creation of HTML

Before HTML, a document author didn't have to care how the document would appear on someone's monitor. It was accepted that appearance was the area which the user will look after.

HTML was evolved in seven versions. One of the requirements for creating usable, nonlinear information system is to make the system easy for non experts to understand and use.

Level 0 HTML

Level 0 HTML was first implemented in 1990. HTML followed the SGML rules of platform independent.

At level 0, HTML offered a platform-independent means of marking data for interchange. The concept was that servers would; store and supply data and clients would retrieve and display it. HTML0.0 was very close to SGML. The only required element was the TITLE element and many older pages still remain that start with a title and then go straight into the text.

Level 0 allowed a <BODY> tag and authors could include addresses, anchors, block quotes, line breaks, headings, horizontal rules, images, lists, paragraphs and preformatted text.

Level 1 HTML

HTML 1.0 was developed in 1992 by Dan Connolly. The idea of an HTML container was added with a HEAD element separated from the BODY Element. Opening and closing tags were required for some elements.

Level 1 also introduced forms, which makes it possible for authors to have input fields on Weir nodes that enable feedback from users and open door to considering interaction through Common Gateway Interface (CGI) scripting.

Did u know? The potential of forms and CGI scripts is still evolving, even with the addition of new form capabilities in HTML4.

2.1.3 HTML and the Advent of Graphics

The Web is about more than text and information, it is also a medium for expressing artistic creativity, data visualization, and optimizing the presentation of information for different audiences with different needs and expectations. The use of graphics on Web sites enhances the experience for users, and W3C has several different and complementary technologies that work together with HTML and scripting to provide the creators of Web pages and Web Applications with the tools they need to deliver the best possible representation of their content.

2.1.4 Advantages of HTML

The HTML is the widely used language for writing web pages. HTML is highly flexible and user friendly. HTML is similar to that of XML, which is used in data storage. HTML has many advantages as shown below:

- HTML is highly flexible
- HTML is supported on almost every browser
- HTML is user friendly
- HTML is an open technology
- HTML is consistent and efficient
- HTML bids the superlative medium for its users
- HTML is easily understandable and does not require any training
- HTML is designed with a feature of interaction between the webpages, which makes it effective
- HTML provides search engine compatible pages
- HTML is easier to maintain and update any site
- HTML does not involve strain on the servers
- For HTML web pages, it takes less time to load the web pages
- HTML validation is another important key factor which increases the web accessibility
- HTML webpage's look and feel attracts larger masses to visit the website

Self Assessment

Fill in the blanks:

1. .... is a markup language which tells the web browser how to display content.
2. A .... is a language that annotates text so that the computer can manipulate the text.
3. .... Language can be described as a combination of several technologies like HTML client-side java script and cascading Style Sheets.
4. ....is a document markup language used mainly bye mathematicians, authors, etc. to typeset their content.
5. .... is used in voice interaction between humans and computer.

2.2 Web Server

A Web server is a program that, using the client/server model and the World Wide Web's Hypertext Transfer Protocol (HTTP), serves the files that form Web pages to Web users (whose computers contain HTTP clients that forward their requests). Every computer on the Internet that contains a Web site must have a Web server program. Two leading Web servers are Apache, the most widely-installed Web server, and Microsoft's Internet Information Server (IIS). Other Web servers include Novell's Web Server for users of its NetWare operating system and IBM's family of Lotus Domino servers, primarily for IBM's OS/390 and AS/400 customers.

Web servers often come as part of a larger package of Internet- and intranet-related programs for serving e-mail, downloading requests for File Transfer Protocol (FTP) files, and building and publishing Web pages. Considerations in choosing a Web server include how well it works with the operating system and other servers, its ability to handle server-side programming, security characteristics, and publishing, search engine, and site building tools that may come with it.

Self Assessment

Fill in the blanks:

6. A .... is a program that, using the client/server model and the World Wide Web's Hypertext Transfer Protocol (HTTP), serves the files that form Web pages to Web users.
7. Every computer on the Internet that contains a .... must have a Web server program.

2.3 Web Browser

Short for *Web browser,* a software application used to locate, retrieve and also display content on the World Wide Web, including Web pages, images, video and other files. As a client/server model, the browser is the client run on a computer that contacts the Web server and requests information. The Web server sends the information back to the Web browser which displays the results on the computer or other Internet-enabled device that supports a browser.

Today's browsers are fully-functional software suites that can interpret and display HTML Web pages, applications, JavaScript, AJAX and other content hosted on Web servers. Many browsers offer plug-ins which extend the capabilities of a browser so it can display multimedia information (including sound and video), or the browser can be used to perform tasks such as videoconferencing, to design web pages or add anti-phishing filters and other security features to the browser.

2.3.1 Definition

A **web browser** is a software application for retrieving, presenting and traversing information resources on the World Wide Web. An *information resource* is identified by a Uniform Resource Identifier (URI) and may be a web page, image, video or other piece of content. Hyperlinks present in resources enable users easily to navigate their browsers to related resources. A web browser can also be defined as an application software or program designed to enable users to access, retrieve and view documents and other resources on the Internet.

2.3.2 Internet Explorer, Firefox and Safari

The two most popular browsers are Microsoft Internet Explorer and Firefox. Other major browsers include Google Chrome, Apple Safari and Opera. While most commonly use to access information on the web, a browser can also be used to access information hosted on Web servers in private networks.

2.3.3 Web Browser History

In computing, the **web browsing history** refers to the list of web pages a user has visited recently—and associated data such as page title and time of visit—which is recorded by web browser software as standard for a certain period of time. Web browser software does this in order to provide the user with a Back button and/or a History list, to go back to pages they have visited previously, rather than relying on the user to remember where they have been on the web.

In addition to the web browser software itself, third-party services can also record a user's web browsing history (completely or partially).

Example: In Google Web History, the clicks of registered users are recorded and stored in individual user histories, each of which are browsable and searchable by that user (this is in addition to the click-tracking Google records for its own internal purposes, such as advertising click tracking). If the user installs the Google Toolbar, all pages that the user visits while logged in to Google on that computer may be recorded as well.

Caution A potential benefit to the user is that they can review - and search through - all of their web browsing history on any computer, but this can have privacy implications.

Self Assessment

Fill in the blanks:

8. .... is a software application used to locate, retrieve and also display content on the World Wide Web, including Web pages, images, video and other files.
9. An .... is identified by a Uniform Resource Identifier (URI) and may be a web page, image, video or other piece of content.
10. .... present in resources enable users easily to navigate their browsers to related resources.
11. Many browsers offer .... which extend the capabilities of a browser so it can display multimedia information.

2.4 Html Tags

HTML tags (otherwise known as "HTML elements"), and their respective attributes are used to create HTML documents so that you can view them in browsers and other user agents.

Notes Not all browsers/user agents support all HTML tags and their attributes, so you should try to test your pages in as many browsers as you can.

Tags are instructions that are embedded directly into the text of a HTML document. Each HTML tag describes that the browser should do something instead of simply displaying the text. In HTML, the tags begin with (<) and end with (>) HTML tags can be of two types. They are

1. Paired Tags
2. Unpaired Tags

2.4.1 Container or Paired Tag

A tag is said to be a paired tag if the text is placed between a tag and its companion tag. In paired tags, the first tag is referred to as *Opening Tag* and the second tag is referred to as *Closing Tag.*

Example: <i>This text is in italics. </i>

Notes Here <i> is called opening tag. and </i> is called closing tag.

2.4.2 Empty or Singular Tags

An unpaired tag does not have a companion tag. Unpaired tags are also known as *Singular* or *Stand-Alone* Tags.

Example: <br> , <hr>, etc. These tags does not require any companion tag.

Task Compare and contrast container tags and singular tags.

Self Assessment

Fill in the blanks:

12. .... are instructions that are embedded directly into the text of a HTML document.
13. In paired tags, the first tag is referred to as Opening Tag and the second tag is referred to as .... Tag.
14. A tag is said to be a .... tag if the text is placed between a tag and its companion tag.
15. Unpaired tags are also known as .... Tags.

2.5 Summary

- A markup language is a language that annotates text so that the computer can manipulate the text.
- Most markup languages are human readable because the annotations are written in a way to distinguish them from the text.
- The HTML is the widely used language for writing web pages.

- A Web server is a program that, using the client/server model and the World Wide Web's Hypertext Transfer Protocol (HTTP), serves the files that form Web pages to Web users (whose computers contain HTTP clients that forward their requests).
- An *information resource* is identified by a Uniform Resource Identifier (URI) and may be a web page, image, video or other piece of content.
- A **web browser** is a software application for retrieving, presenting and traversing information resources on the World Wide Web.
- **Web browsing history** refers to the list of web pages a user has visited recently.
- Tags are instructions that are embedded directly into the text of a HTML document.

2.6 Keywords

DHTML: Dynamic Hyper Text Markup Language can be described as a combination of several technologies like HTML client-side java script and cascading Style Sheets.

HTML tags: Tags are instructions that are embedded directly into the text of a HTML document.

Html: HTML is a **markup language**. It tells the web browser how to display content.

LaTeX: A document markup language used mainly bye mathematicians, authors, etc. to typeset their content. It is suitable for representing mathematical formulas.

Markup language: Language that annotates text so that the computer can manipulate the text.

VoiceXML*:* Used in Voice interaction between humans and computer

Web browser: It is a software application for retrieving, presenting and traversing information resources on the World Wide Web.

Web server: A Web server is a program that, using the client/server model and the World Wide Web's Hypertext Transfer Protocol (HTTP), serves the files that form Web pages to Web users

2.7 Review Questions

1. Discuss the history of HTML.
2. Discuss some of the text manipulating tags used in HTML.
3. List out various advantages and limitation of HTML.
4. Differentiate between singular and paired tags.
5. How to work on web server?
6. What is advantage of HTML?
7. Define the web browser.
8. What is markup language?
9. Make distinction between DHTML and XHTML.
10. Explain the creation of HTML.

Notes

Answers: Self Assessment

1. HTML
2. markup language
3. Dynamic Hyper Text Markup
4. LaTeX
5. VoiceXML
6. Web server
7. Web site
8. Web browser
9. information resource
10. Hyperlinks
11. plug-ins
12. Tags
13. Closing
14. Paired
15. Singular or Stand-Alone

2.8 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://rajeshstutorials.blogspot.in

http://webdesign.about.com/

http://www.rapidprogramming.com/

http://www.w3.org

https://developer.mozilla.org/en-US/docs/HTML/Introduction

Sarabjit Kumar, Lovely Professional University

Notes

Unit 3: HTML Command and Structure

CONTENTS

Contd....

Objectives

After studying this unit, you will be able to:

- Understand the uses of HTML commands
- Explain the structure of HTML program
- Describe the text formatting tags
- Define text styles
- Explain the effect of text

Introduction

Hypertext Markup Language (**HTML**) is the main markup language for creating web pages and other information that can be displayed in a web browser.

HTML is written in the form of HTML elements consisting of *tags* enclosed in angle brackets (like <html>), within the web page content. HTML tags most commonly come in pairs like <h1> and </h1>, although some tags, known as *empty elements*, are unpaired, for example <img>. The first tag in a pair is the *start tag*, the second tag is the *end tag* (they are also called *opening tags* and *closing tags*). In between these tags web designers can add text, tags, comments and other types of text-based content.

The purpose of a web browser is to read HTML documents and compose them into visible or audible web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages.

3.1 Html Commands

Tags typically occur in begin-end pairs. These pairs are in the form tag> ... </tag> Where the <tag> indicates the beginning of a tag-pair, and the </tag> indicates the end. (The three dots indicate an arbitrary amount of content between the tags.). These pairs define **containers**. Any content within a container has the rules of that container applied to it.

Example: The text within a "boldface container" would be boldfaced.

Similarly, paragraphs are defined using a "paragraph container."

Some commands do not consist of a begin and end tag, but just of a single tag. In HTML, this is just a begin tag:<tag>

3.1.1 Html Tags

These are the tags that tell a web browser where the HTML part in your document begins and ends.

```
<html>
</html>
```

Inside the "html" container, we have the "head" and the "body" container:

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

The **"body"** contains the actual content of your web page.

The **"head"** contains all of the document's header information like the web document's title and information about the document itself. This is an important point for search engines.

The container **"title"** is placed within the head structure. Between the title tags, you should have the title of your document. This will appear at the top of the browser's title bar, and also appears in the history list. Finally, the contents of the title container go into your bookmark file, if you create a bookmark to a page.

Also, the head contains **meta** information about the document, most importantly the character encoding that is used.

Notes The **encoding** "ISO-8859-1" is used for English, French, Spanish, German and other western European languages.

Here you see an example of a "head" container with a "title" element and a "meta" element denoting the encoding:

```
<head>
<title>This is my very first HTML document</title>
<meta http-equiv="content-type" content="text/html;
charset=ISO-8859-1">
</head>
```

Again, you can just copy the "meta" element as you see it above into your document.

The "**body"** comes after the head structure. Between the body tags, you find all of the stuff that gets displayed in the browser window. All of the text, the graphics, and links, and so on - these things occur between the body tags.

The strict variant of HTML 4.01 requires that any content inside the body is within a further set of tags (if you use the transitional variant of HTML 4.01, this is not necessary). For text, you can use "p" (the paragraph tag). A complete page would then look like this as shown in example:

Example: Writing a paragraph in HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/ukt.dtd">
<html>
```

Notes

```
<head>
<title>This is my very first HTML document</title>
<meta http-equiv="content-type" content="text/html;
charset=ISO-8859-1">
</head>
<body>
```

<p><h1> Here is where the actual content of the page goes.

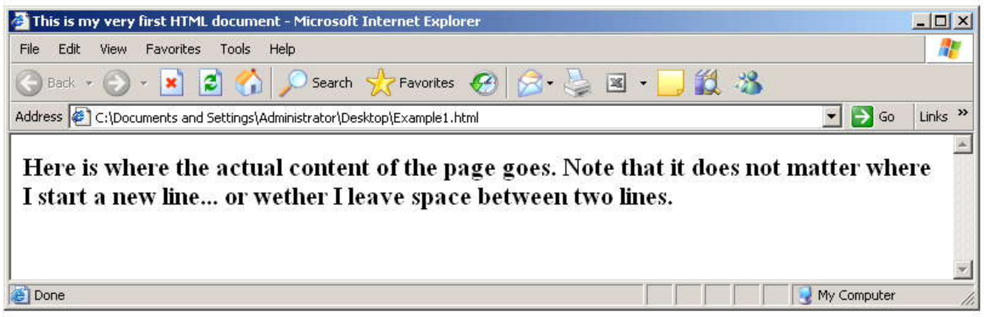
Note that it does not matter where I start a new line...

or wether we leave space between two lines. </p></h1>

```
</body>
</html>
```

This will result in the following page being displayed in your browser as shown in Figure 3.1:

Figure 3.1: Output


This is my very first HTML document - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address C:\Documents and Settings\Administrator\Desktop\Example1.html
Go Links
Done
My Computer

Here is where the actual content of the page goes. Note that it does not matter where I start a new line... or wether I leave space between two lines.

If you want to leave yourself notes in an HTML document, but don't want those notes to show up in the browser window, you need to use the comment tag. To do that, you would do the following:

<!— This is a comment. —>

Steps to run this example

1. Save the above example as HTMLExample.html.
2. Open the browser.
3. Open the file by browsing it
4. Get the output

3.1.2 Headers

Heading structures are most commonly used to set apart document or section titles.

There are six levels of headings, from heading 1 through heading 6. Heading1 (h1) is "most important" and heading 6 (h6) is "least important." Bydefault, browsers will display the six heading levels in the same font, with the point size decreasing as the importance of the heading decreases. Here are two examples:

```
<h1>Heading 1: The largest</h1>
<h6>Heading 6: The smallest</h6>
```

3.1.3 Paragraphs

To create an empty line between two blocks of text, you need to label those text blocks (or paragraphs) with the paragraph marker "p". So, surround your paragraphs with <p> and </p> like we have done in our example page. Actually, in HTML you do not need the closing tag for "p", you can create an empty line after a block by just writing <p>.

Caution You do not actually need to leave a blank line between the two paragraphs in your source code.

Here is an example:

<p>This is the first paragraph. Isn´t it a nice paragraph? It has lots of words in it. It´s such a nice paragraph that I think it should never end.</p>

<p>This is the second paragraph. It´s also a very nice paragraph

but maybe not as nice as that first one was.</p>

3.1.4 Preformatted Text

The HTML pre element inserts a block of preformatted text. This is exclusively a visual element, as the semantic of the text remains the same with or without it. In fact, its specifications are exclusively for visual user agents.

To render this element, we should follow these guidelines:

- White space may be left intact, which means that consecutive spaces shouldn't be stripped.
- Text may be rendered with a fixed-pitch font, where all characters have the same width.
- Automatic word-wrap may be disabled, so text lines won't be splitted unless a line break is found.
- The bidirectional processing must not be disabled.

3.1.5 Boldface and Italics

Using bold text will display your text in a thicker font making one or more words really stand out from the rest and will tend to draw the reader's attention to these words. This has a variety of applications and is largely a matter of taste.

Did u know? Bold text is sometimes used for list headers such as in this list of font styles. To create bold text, place the desired text within the <b>...</b> tags.

Example: <b>Your bold text goes here.</b>

The italics font style *slants the text to the right* and thus can also be used to draw special attention to one or more words. You may wish to use italics instead of bold when the bold font style is too "loud" for your purposes. Italics has a variety of other applications and the use of it is largely a matter of taste. The italics are often formally applied to the titles of newspapers, magazines and books, such as when one wants to mention *The Times of India*. To display text in italics using HTML, place the desired text within the <i>...</i> tags.

Example: <i>Your italics text goes here.</i>

3.1.6 Physical Tags

A *physical tag* controls how the characters are formatted. For instance, you might display some characters as bold or italic. Table 3.1 displays some common physical character tags.

Table 3.1: Common Physical Character Tags

Tag	Description	Renders as
<b>	bold	displays text as bold
<big>	big	displays text in a big font
<i>	italics	displays text as italic
<s> or <strike> *	strike	displays text with a line through it
<small>	small	displays text in a small font
<sub>	subscript	displays the text as subscript — text that displays below the baseline of the text
<sup>	superscript	displays the text as superscript — text that has baseline above the baseline of the rest of the text
<tt>	teletype	displays the text with fixed-width font
<u>	underline	underlines the text

3.1.7 Logical Tags

A *logical character tag* describes how the text is being used, not necessarily how it is formatted.

Table 3.2 displays common examples logical character tags.

Table 3.2: Common Logical Character Tags

Tag	Description	Renders as
<cite>	citation	emphasizes the text in italics.
<code>	code sample	Displays some characters as code usually in Courier font (i.e., fixed-width font)
<del>	deleted text	displays text with a line through it; renders differently in Netscape and Internet Explorer
<dfn>	definition	italics; renders differently in Netscape and Internet Explorer
<em>	emphasis	emphasizes the text in some way usually as italic.
<ins>	inserted text	underlined; renders differently in Netscape and Internet Explorer
<kbd>	code sample	fixed-width font
<samp>	code sample	fixed-width font
<strong>	strong	Text is emphasized more strongly than usually as bold.
<var>	program variable	italics

Task Compare and contrast physical tags and logical tags.

Notes

Self Assessment

Fill in the blanks:

1. HTML .... tell a web browser where the HTML part in your document begins and ends.
2. The ".........................." contains all of the document's header information like the web document's title and information about the document itself.
3. The HTML pre element inserts a block of .... text.
4. A .... tag controls how the characters are formatted.
5. A ....tag describes how the text is being used, not necessarily how it is formatted.
6. The .... font style slants the text to the right and thus can also be used to draw special attention to one or more words.

3.2 Structure of an HTML Program

A HTML document is basically separated in two parts: the head (HTML head tag) and the body (HTML body tag). We also add a *Document type declaration* on top of it to get the basic document structure and HTML version

3.2.1 Basic Structure

The structure of a HTML document is shown below.

Structure of HTML Program

```
<! DOCTYPE …> version information
<HTML>
  <HEAD>
 ...information about document, scripts, styles….
  </HEAD>
  <BODY>
  ...visible content of document….
  </BODY>
</HTML>
```

<HTML> tag: The html tag acts as a container for the whole document. Every character in the document should be in between the html start and end tags. The html tag can also be used to define the language of the contained document through the "lang" attribute. The content of the html tag is divided in two parts using the head (HTML head tag) and the body (HTML body tag).

<HEAD> tag: This section is the document's head. All the information contained in the document's head is loaded first, before any other thing in the document, as it's defined before the body segment. It includes tags like title, script, style, meta and so on.

<BODY> tag: This is the document's body. The body is the container for the visual part of a document. All the things written here will be shown when the document is rendered. Most of the tags in HTML can be inserted in the body section (inside the HTML body tag) and will take care of the visual aspects of the document.

3.2.2 !DOCTYPE Declaration

Every well written HTML document begins with a basic declaration that defines what type of document it is. This declaration is made using the!DOCTYPE tag and is to be written at the beginning of the document. It tells the processing agent and HTML version. Sample is shown below.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
```

Self Assessment

Fill in the blanks:

7. A HTML .... is basically separated in two parts: the head and the body .
8. Every well written HTML document begins with a basic declaration that defines what type of document it is and this declaration is made using the .... tag.

3.3 Text Formatting Tags

After any length of time on the Internet, you'll notice that a Web page is made up of more than just plain words on a screen. There are headlines, paragraphs, graphics, colors and much more. It's a lively place to be.

Our next tags—headline, paragraph, line break and horizontal rule—will help us make our current page a lot more exciting.

3.3.1 Paragraph Breaks <P>

To add space between paragraphs you use the paragraph tag:

<p>…</p>

This is a container tag and must have a beginning and an ending.

To add a single line of space, you use break tag:

<br>

This is an empty tag and stands alone. You can use the <br> tag to insert one or more blank lines.

3.3.2 Horizontal Rules <HR>

To create a horizontal line on your page you use the empty tag:

<hr>

3.3.3 Heading Styles

In HTML, bold copy is created by using the headline tag. There are six levels of headlines, ranging from **<h1>…</h1> to <h6>…</h6>**.

Here is an example of the code for *all* the headline sizes:

```
<h1>Level 1 Headline</h1>
<h2>Level 2 Headline</h2>
```

Notes

```
<h3>Level 3 Headline</h3>
<h4>Level 4 Headline</h4>
<h5>Level 5 Headline</h5>
<h6>Level 6 Headline</h6>
```

Self Assessment

Fill in the blanks:

9. .... tag is used to create a horizontal line on your page you use the empty tag.
10. In HTML, bold copy is created by using the .... tag.

3.4 Text Styles

The following HTML tags are used to format the appearance of the text on your web page. This can jazz up the look of the web page, *however,* too much variety in the text style can also look displeasing.

3.4.1 Bold

Refer 3.1.5

3.4.2 Italics

Refer 3.1.5

3.4.3 Underline <u> </u>

Again, the same as underline in a word processor.

Did u know? html links are already underlined and don't need the extra tag.

3.4.4 Centering (Text, Images, etc.) <center> </center>

A useful tag, as it says, it makes everything in between the tags centred (in the middle of the page).

3.4.5 Spacing

Using &nbsp; is helpful in situations where you just need one or two spaces in between characters or words for stylistic or aesthetic effect.

Example: If you wanted a particular part of your text to read with two spaces after a period - as in "Hello. How are you?" – you'd have to code every extra period in as follows: "Hello. &nbsp;How are you?"

3.4.6 Line Breaks <BR>

Refer 3.3.1

3.4.7 Controlling Font Sizes and Color

Font Size Change - <font size="+/-?"> </font>

For an immediate change of font size with respect to the font size preceding it, this tag increase or decreases the size of the font by the number you specify. Eg: <font size="-1">Some Text</font>

Font Colour - <font color="#??????"> </font>

Change the colour of a few words or a section of text. The 6 question marks represent the hex color code,

3.4.8 Preformatted Text <PRE>

Refer 3.1.4

3.4.9 Text Highlighting Tags

Emphasis - <em> </em>

Used to emphasize text, which usually appears in italics, but can vary according to your browser.

Strong Emphasis - <strong> </strong>

Used to emphasize text more, which usually appears in bold, but can vary according to your browser.

3.4.10 Text Alignment

The code to align text in HTML for a post or single page is <P ALIGN=direction>.When placing this code onto the page, you simply put where you'd like the text to be justified into where it says "direction" in the example.

Task Illustrate how to apply bold, italics, and **control Font Sizes and Color.**

Self Assessment

Fill in the blanks:

11. Using .... is helpful in situations where you just need one or two spaces in between characters or words for stylistic or aesthetic effect.
12. The tag .... change the colour of a few words or a section of text.
13. The code to align text in HTML for a post or single page is ....

3.5 Text Effect

There are a couple of simple text effects that can be produced just by using HTML tags. If you need more complicated effects you could use animated GIF images or JavaScript.

Notes

Caution Be aware that any animated or flashing objects can make pages difficult to read for people with certain types of vision impairment.

3.5.1 Header Tags

Refer 3.1.2

3.5.2 Italics and Bolding

Refer 3.1.5

3.5.3 Flashing Text

This is a non-standard tag (it only works with Netscape browsers) and can be **very** irritating if used on large areas of text, so use it sparingly.

<blink> ... </blink> Place around text to make it flash. (If you're not using Netscape Navigator you're really seeing an image that looks like the blink tag effect.)

The code could look **like this**:

<p>Place around text to make it <blink>flash</blink>.</p>

3.5.4 Moving Text

Another non-standard tag that can be as irritating as the <blink> tag is <marquee>. This can be used to create a moving text effect, similar to some "matrix" type LED and illuminated signs.

Notes There are quite a few options to control the appearance of text formatted with this tag, the main ones are listed below. This tag works on Internet Explorer and later versions of Netscape Navigator.

In early versions of Netscape Navigator you will see the text as a normal paragraph, it will not scroll.

<marquee ... >	This tag starts the Marquee, the following four attributes should be put in this tag before the closing >.
behavior="..."	This attribute can be "scroll", "slide" or "alternate". Scroll makes the text scroll all the way across, off the edge and then starts again. Slide makes the text scroll in like "scroll" but it stops when it reaches the far edge. Alternate makes the text bounce back and forth between its' edges.
bgcolor="#..."	This will change the colour of the background using the usual colour conventions.
loop="6"	Specifies the number of times the effect will run, e.g. six times, or you can specify "infinite" or "-1" to loop continuously.
width="300"	Specifies the width of the marquee effect in pixels, or you can specify it as a percentage of the space available, e.g. "40%".
Message Text	After the opening <marquee ... > tag type the text that will scroll.
</marquee>	This is the end tag.

The code could look **like this**:

<marquee behavior="scroll" bgcolor="#00CCCC" loop="-1" width="40%">Demonstration of a Marquee tag.</marquee>

Self Assessment

Fill in the blanks:

14. .... tag is a non-standard tag and can be **very** irritating if used on large areas of text, so use it sparingly.
15. .... tag is used to create a moving text effect, similar to some "matrix" type LED and illuminated signs.

3.6 Summary

- A HTML document is basically separated in two parts: the head (HTML head tag) and the body (HTML body tag).
- HTML tags that tell a web browser where the HTML part in your document begins and ends.
- There are a couple of simple text effects that can be produced just by using HTML tags.
- Heading structures are most commonly used to set apart document or section titles.
- Using &nbsp; is helpful in situations where you just need one or two spaces in between characters or words for stylistic or aesthetic effect.
- In HTML, bold copy is created by using the headline tag.
- The body is the container for the visual part of a document.
- Every well written HTML document begins with a basic declaration that defines what type of document it is.

3.7 Keywords

Body: The body is the container for the visual part of a document.

Head: The **"head"** contains all of the document's header information like the web document's title and information about the document itself. This is an important point for search engines.

HTML pre element: The HTML pre element inserts a block of preformatted text.

HyperText Markup Language: **HTML** is the main markup language for creating web pages and other information that can be displayed in a web browser.

Logical Tag: A logical character tag describes how the text is being used, not necessarily how it is formatted.

Physical tag: It controls how the characters are formatted.

Tags: Html tags tell a web browser where the HTML part in your document begins and ends.

Title: The container **"title"** is placed within the head structure.

3.8 Review Questions

1. What are HTML commands? How they are used?
2. Discuss various advantages and limitations of HTML.
3. What is the structure of an HTML program?
4. Describe text formatting tags with the help of example.
5. What do you mean by text effect?
6. What are the different types of header tags? Explain with example.
7. What do you mean by preformatted text?
8. What is Preformatted Text? Discuss.
9. Discuss the guidelines used to render HTML pre element.
10. Explain the tags used for Text Formatting.

Answers: Self Assessment

1.	tags	2.	head
3.	preformatted	4.	physical
5.	logical character	6.	italics
7.	document	8.	!DOCTYPE
9.	<HR>	10.	headline
11.	&nbsp;	12.	Font Colour
13.	<P ALIGN=direction>	14.	<blink>
15.	<marquee>		

3.9 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://www.goodellgroup.com

http://www.ironspider.ca/

http://www.scriptingmaster.com

http://www.simplehtmlguide.com

Unit 4: HTML List

Notes

CONTENTS

Objectives

After studying this unit, you will be able to:

- Describe about HTML Tables
- Explain how HTML Tables are created
- Explain the steps involved in creating HTML Tables
- Describe about Linking Documents
- Describe about Creating Link Lists

Introduction

Lists commonly are found in documents, including web pages. They are an easy and effective way to itemize such things as elements, components, or ingredients.

Words or phrases which need to be set apart from the rest of the body of text can be emphasized with a "bullet" (a heavy dot used for calling attention to a particular section of text). An *empty* tag called a "list" tag is used to do this:

<LI>: creates a bullet in front of text which is to be set apart for emphasis and causes all text after it to be indented, either until another list tag is detected or until the end of the list is reached. It is used to itemize elements of "unordered" and "ordered" lists.

Notes A <BR> tag is *not* inserted at the end of an item beginning with a <LI> tag, as a *line break* automatically occurs at that point.

Since a list tag is an *empty* tag (that is, there is no negating counterpart </LI>), and since it indents the text following it, it cannot be alone; otherwise, the entire remainder of the document would be indented.

4.1 HTML List Classes

The HTML List classes allow you to easily create lists within your HTML pages. These classes provide methods to get and set various attributes of the lists and the items within the lists.

In particular, the parent class HTML List provides a method to produce a compact list that displays items in as small a vertical space as possible.

- Methods for HTML List include:
  - Compact the list
  - Add and remove items from the list
  - Add and remove lists from the list (making it possible to nest lists)
- Methods for HTML List Item include:
  - Get and set the contents of the item
  - Get and set the direction of the text interpretation
  - Get and set the language of the input element

Use the subclasses of HTML List and HTML List Item to create your HTML lists:

- OrderedList and OrderedListItem
- UnorderedList and UnorderedListItem

4.1.1 OrderedList and OrderedListItem

Use the OrderedList and OrderedListItem classes to create ordered lists in your HTML pages.

- Methods for OrderedList include:
  - Get and set the starting number for the first item in the list
  - Get and set the type (or style) for the item numbers
- Methods for OrderedListItem include:
  - Get and set the number for the item
  - Get and set the type (or style) for the item number

Did u know? By using the methods in OrderedListItem, you can override the numbering and type for a specific item in the list.

4.1.2 UnorderedList and UnorderedListItem

Use the UnorderedList and UnorderedListItem classes to create unordered lists in your HTML pages.

- Methods for UnorderedList include:
  - Get and set the type (or style) for the items
- Methods for UnorderedListItem include:
  - Get and set the type (or style) for the item

Task Illustrate the methods for UnorderedList and UnorderedListItem with example.

Self Assessment

Fill in the blanks:

1. The .... classes allow you to easily create lists within your HTML pages.
2. The .... OrderedList and OrderedListItem are used to create ordered lists in your HTML pages.
3. By using the methods in ...., you can override the numbering and type for a specific item in the list.

4.2 HTML List Types

HTML provides three different types of lists to choose from when building a page, including unordered, ordered, and definition lists. Unordered lists are for lists of items where order isn't of important. While ordered lists place strong importance on the order of items. In the case where there is a list of terms and descriptions, perhaps for a glossary, definition lists are available.

4.2.1 Unordered List

Unordered lists are purely a list of related items, in which their order does not matter nor do they have a numbered or alphabetical list element. Creating an unordered list in HTML is accomplished using the unordered list, ul, block level element. Each list item within an unordered list is individually marked up using the list item, li, block level element.

By default most browsers represent each list item with a solid dot.

Caution The solid dot is referred to as the list item element and can be changed using CSS.

Example:

```
<ul>
  <li>Tamilnadu</li>
  <li>Uttar  Pradesh</li>
  <li>West-Bengal</li>
</ul>
```

4.2.2 Ordered List

The ordered list element, ol, works just like the unordered list element, including how each individual list item is created. The main difference between an ordered list and an unordered list is that with an ordered list the order of which items are represented is important. Instead of showing a dot as the default list item element, an ordered list uses numbers. Using CSS, these numbers can then be changed to letters, Roman numerals, and so on.

Example:

```
<ol type="1">
  <li>Tamilnadu</li>
  <li>Uttar Pradesh</li>
  <li>West-Bengal</li>
</ol>
```

4.2.3 Definition List

Creating a definition list in HTML is accomplished using the dl element. Instead of using the li element to mark up list items, the definition list actually requires two elements: the definition term element, dt, and the definition description element, dd.

A definition list may contain numerous terms and descriptions, one after the other. Additionally, a definition list may have multiple terms per description as well as multiple descriptions per term. A single term may have multiple meanings and warrant multiple definitions. In comparison, a single description may be suitable for multiple terms.

```
<dl>
<dt>study</dt>
<dd>the devotion of time and attention to acquiring knowledge on an
academic subject, esp. by means of books</dd>
<dt>design</dt>
<dd>a plan or drawing produced to show the look and function or workings of
a building, garment, or other object before it is built or made</dd>
<dd>purpose, planning, or intention that exists or is thought to exist
behind an action, fact, or material object</dd>
<dt>business</dt>
<dt>work</dt>
<dd>a person’s regular occupation, profession, or trade</dd>
</dl>
```

Self Assessment

Fill in the blanks:

4. .... lists are purely a list of related items, in which their order does not matter nor do they have a numbered or alphabetical list element.
5. By default most browsers represent each list item with a ....
6. The solid dot is referred to as the list item element and can be changed using ....
7. .... lists place strong importance on the order of items.
8. Creating a definition list in HTML is accomplished using the .... element.
9. The .... actually requires two elements: the definition term element, dt, and the definition description element, dd.

4.3 Graphics to HTML Document

Notes

One of the best ways to create an impact with your web page is to add some graphics. Graphics can add color, depth and sparkle to a web page and can serve to catch the eye and retain viewer interest.

A *graphic* picture can be worth a thousand words—and even more than that if it is animated. Graphics and icons can turn a dull, drab web page into a vibrant, exciting one. An image which is displayed on a web browser is referred to as an "inline image."

Did u know? To place a graphic image somewhere on a page, an *empty* "image" tag is used.

4.3.1 IMG Attributes

<IMG> indicates that an image—such as a photograph, icon, animation, cartoon, or other graphic—is to be displayed at that location.

The <IMG> tag should contain within it further parameters as part of the command:

- ***SRC= "URL/graphic.gif or .jpg":*** contains the URL (Uniform Resource Locator or web address) and name of the graphic image file, such as graphic.gif or graphic.jpg. Most commonly, the photograph, icon, or other graphic is a "gif" (Graphics Interchange Format image) or a "jpg" (Joint Photographic Experts Group image), both of which are recognized by most browsers. Some browsers also will recognize a "bmp" (Bitmap image) and a "tif" or "tiff" (Tag Image File Format image).

  Usually, the location source of the graphic file is in an adjacent directory such as "graphics," or it possibly might be in the same directory. Assuming the image is a .jpg image, if the graphic is in an adjacent "graphics" directory, the tag would read: <IMG SRC= "graphics/graphic.jpg">. If the image is located within the same directory as the document, the tag would read simply: <IMG SRC="graphic.jpg">. If the location of the image is somewhere else on the web, the tag might read something like this: <IMG SRC= "http://www.tedmontgomery.com/tutorial/graphics/graphic.gif">.

- **ALIGN="LEFT" | "RIGHT" | "TOP" | "TEXTTOP" | "MIDDLE" | "ABSMIDDLE" | "BASELINE" | "BOTTOM" | "ABSBOTTOM"**: places the graphic image at a specified position, in relation either to the page margins or to the text. (Some browsers will not recognize all of these parameters.)

  "LEFT" aligns the image with the left margin of the page and allows text to wrap around the right side of the image. "RIGHT" aligns the image with the right margin of the page and allows the text to wrap around the left side of the image. Note: The only way to center a graphic horizontally on a page is to use <CENTER> & </CENTER> tags around the <IMG SRC> tag. However, centering a graphic in this manner will prevent text from being wrapped around either side of it. Also, any ALIGN="RIGHT" or ALIGN="LEFT parameter within the <IMG> tag will *override* the effect of the centering tags. "TOP" aligns the top of the image with the top of the tallest item in the line. "TEXTTOP" aligns the top of the image with the top of the tallest text in the line; usually, but not always, the same as the "TOP" parameter. "MIDDLE" aligns the middle of the image with the baseline of the current line. "ABSMIDDLE" aligns the middle of the image with the middle of the current line. "BASELINE" aligns the bottom of the image with the baseline of the current line. "BOTTOM" is the same as the "BASELINE" parameter. "ABSBOTTOM" aligns the bottom of the image with the bottom of the current line; usually, but not always, the same as the "BASELINE" or "BOTTOM" parameter.

- ***WIDTH="W":*** defines the width "W" of the image in pixels.
- ***HEIGHT="H":*** defines the height "H" of the image in pixels.

Notes The WIDTH and HEIGHT parameters, when included in the <IMG SRC> tag, cause each graphic image to be loaded more quickly into a browser. For a page with many graphics, this significantly can reduce the time it takes for the page to be displayed fully. The width and height of a graphic image can be determined by using graphic software, such as Paint Shop Pro or Adobe Photoshop. A proportionally *smaller* space can be made into which an image can fit (e.g., a 40 × 60 image into a 20 × 30 space) and still maintain a distinct appearance. However, if an image is made to fit into a proportionally *larger* space (e.g., a 40 × 60 image into a 50 × 75 space), it will appear rough and grainy.

- ***BORDER="B":*** creates a border around the image, with a uniform width of "B" in pixels. (In case the image is incorporated as a hyperlink, the unvisited, active, and visited colors of the border will be the same as that of the other text hyperlinks on the page.) The *default* setting is BORDER= "2" (that is, a border 2 pixels wide).

Caution BORDER="0" causes *no* border to be present around the hyperlink image.

- ***HSPACE="H":*** creates a space, with width "H" in pixels, between the image and any text immediately to the right and/or left of it. (HSPACE means "horizontal space.")
- ***VSPACE="V":*** creates a space, with height "V" in pixels, between the image and any text immediately above and/or below it. (VSPACE means "vertical space.")
- ***ALT="alternate description":*** supplies a description of the image, which will be displayed instead of the image on non-graphical browsers. On typical graphical browsers, this description will appear before the image has loaded and also when the arrow is placed anywhere on the image.
- ***TITLE="title":*** same function as the ALT tag, which is not recognized by some browsers.
- ***ISMAP:*** indicates a server-side image map.
- ***USEMAP:*** indicates a client-side image map.

Task Illustrate the use of <IMG> tag with example.

Self Assessment

Fill in the blanks:

10. An image which is displayed on a web browser is referred to as an "..........................."
11. ....tag indicates that an image—such as a photograph, icon, animation, cartoon, or other graphic—is to be displayed at that location.
12. The only way to center a graphic horizontally on a page is to use <CENTER> & </CENTER> tags around the .... tag.
13. "..........................." aligns the bottom of the image with the bottom of the current line.

14. .... indicates a server-side image map.
15. .... indicates a client-side image map.

4.4 Summary

- *Lists* commonly are found in documents, including web pages.
- The HTML List classes allow you to easily create lists within your HTML pages.
- HTML List provides a method to produce a compact list that displays items in as small a vertical space as possible.
- HTML provides three different types of lists to choose from when building a page, including unordered, ordered, and definition lists.
- Ordered lists place strong importance on the order of items.
- Unordered lists are purely a list of related items, in which their order does not matter nor do they have a numbered or alphabetical list element.
- A definition list may contain numerous terms and descriptions, one after the other.
- One of the best ways to create an impact with your web page is to add some graphics.

4.5 Keywords

<IMG>: <IMG> indicates that an image—such as a photograph, icon, animation, cartoon, or other graphic—is to be displayed at that location.

DL: Definition lists, created using the DL element, generally consist of a series of term/definition Pairs.

HTML List classes: It allows you to easily create lists within your HTML pages.

Inline image: An image which is displayed on a web browser is referred to as an "inline image".

List tag: An empty tag called a "list" tag is used to do itemize elements of "unordered" and "ordered" lists.

OL: An ordered list, created using the OL element, should contain information where order should be emphasized.

Ordered List Item: It allows you to override the numbering and type for a specific item in the list.

Unordered lists: Unordered lists are for lists of items where order isn't of important.

4.6 Review Questions

1. On what information we should emphasize while preparing an order list?
2. Discuss the methods for HTML List.
3. Discuss the methods to create ordered lists, unordered lists, and nested lists.
4. Discuss IMG Attributes.
5. Explain with examples about graphic image alignment parameters in the HTML.
6. Explain the three different types of HTML lists.

7. Differentiate between UnorderedList and OrderedListItem.
8. Discuss the methods for Methods for UnorderedListItem.
9. How can you insert an image in the current page?
10. What are the classes of HTML List? Discuss.

Answers: Self Assessment

1.	HTML List	2.	Classes
3.	OrderedListItem	4.	Unordered
5.	solid dot	6.	CSS
7.	Ordered	8.	Dl
9.	definition list	10.	inline image
11.	<IMG>	12.	<IMG SRC>
13.	ABSBOTTOM	14.	ISMAP
15.	USEMAP		

4.7 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://learn.shayhowe.com/html-css/ordered-unordered-definition-lists

http://publib.boulder.ibm.com/infocenter/iadthelp/v7r5/index.jsp?topic=/com.ibm.etools.iseries.toolbox.doc/htmllist.htm

http://www.tedmontgomery.com/

http://www.tedmontgomery.com/tutorial/graphics.html

Unit 5: Creating Tables

CONTENTS

Objectives

After studying this unit, you will be able to:

- Describe about HTML Tables
- Explain how HTML Tables are created
- Explain the steps involved in creating HTML Tables
- Describe about Linking Documents
- Describe about Creating Link Lists

Introduction

In this unit, we describe the statements for creating and updating tables. We assume that the user knows about the data which has to be stored and what the structure of the data is, i.e., what tables are to be created and what the appropriate columns are.

Notes

5.1 HTML Tables

Tables are defined with the <table> tag. A table is divided into rows (with the <tr> tag), and each row is divided into data cells (with the <td> tag). The letters td stands for "table data," which is the content of a data cell. A data cell can contain text, images, lists, paragraphs, forms, horizontal rules, tables, etc.

Example:

```
<table border="1">
<tr>
<th>Heading</th>
<th>Another Heading</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

Output

Heading	Another Heading
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

Self Assessment

Fill in the blanks:

1. Tables are defined with the .... tag.
2. The letters .... stands for "table data," which is the content of a data cell.
3. A .... can contain text, images, lists, paragraphs, forms, horizontal rules, tables, etc.
4. A table row is define by .... tag.

5.2 Creating Tables

The basic structure of an HTML table consists of the following tags:

- *Table tags:* **<TABLE> </TABLE>**
- *Row tags:* **<TR> </TR>**
- *Cell tags:* **<TD> </TD>**

Constructing an HTML table consists of describing the table between the beginning table tag, **<TABLE>**, and the ending table tag, **</TABLE>**. Between these tags, you then construct each row and each cell in the row. To do this, you would first start the row with the beginning row tag,

<TR>, and then build the row by creating each cell with the beginning cell tag, **<TD>**, adding the data for that cell, and then closing the cell with the ending cell tag, **</TD>**. When you finish all of the cells for a row, you would then close the row with the ending row tag, **</TR>**.

Notes For each new row, you would repeat the process of beginning the row, building each cell in the row, and closing the row.

Example: The following table is an example of a basic table with three rows and two columns of data.

Data 1	Data 2
Data 3	Data 4
Data 5	Data 6

The codes that generated this table will look like this:

```
<TABLE>
 <TR>
  <TD>Data 1</TD>
  <TD>Data 2</TD>
 </TR>
 <TR>
  <TD>Data 3</TD>
  <TD>Data 4</TD>
 </TR>
 <TR>
  <TD>Data 5</TD>
  <TD>Data 6</TD>
 </TR>
</TABLE>
```

This table contains no border, title, or headings. If you wish to add any of these elements to your table, you need to include additional HTML codes.

5.2.1 Tables and the Border Attribute

If you do not specify a border attribute the table will be displayed without any borders. Sometimes this can be useful, but most of the time, you want the borders to show.

To display a table with borders, you will have to use the border attribute:

```
<table border=”1">
<tr>
<td>Row 1, cell 1</td>
<td>Row 1, cell 2</td>
</tr>
</table>
```

5.2.2 Headings in a Table

Headings in a table are defined with the <th> tag.

```
<table border=”1">
```

Notes

```
<tr>
<th>Heading</th>
<th>Another Heading</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

Output

Heading	Another Heading
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

5.2.3 Empty Cells in a Table

Table cells with no content are not displayed very well in most browsers.

```
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td></td>
</tr>
</table>
```

Output

row 1, cell 1	row 1, cell 2
row 2, cell 1	

Note that the borders around the empty table cell are missing (NB! Mozilla Firefox displays the border).

To avoid this, add a non-breaking space (&nbsp;) to empty data cells, to make the borders visible:

```
<table border="1">
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
```

```
<td>&nbsp;</td>
</tr>
</table>
```

Output

row 1, cell 1	row 1, cell 2
row 2, cell 1	

5.2.4 Table Tags

Table 5.1 is showing the table tags.

Tale 5.1: Table Tags

Tag	Description
<table>	Defines a table
<th>	Defines a table header
<tr>	Defines a table row
<td>	Defines a table cell
<caption>	Defines a table caption
<colgroup>	Defines groups of table columns
<col>	Defines the attribute values for one or more columns in a table
<thead>	Defines a table head
<tbody>	Defines a table body
<tfoot>	Defines a table footer

Task Illustrate the use of various table tags.

Self Assessment

Fill in the blanks:

5. When you finish all of the cells for a row, you would then close the row with the ending row tag, ...

6. If you do not specify a .... attribute the table will be displayed without any borders.

7. Headings in a table are defined with the .... tag.

5.3 Linking Document

Once you have the ability to create HTML pages, you'll want to learn how to create links between them, so that you can start building a site. Links are the essence of HTML — they are what makes it unique.

What makes the web so effective is the ability to define links from one page to another, and to follow links at the click of a button. A single click can take you right across the world!

5.3.1 Links

Links are defined with the <a> tag.

Let's create a link to the page defined in the file "anand.html": This link to

```
  <a href="Example1.html">upendra's homepage</a>.
```

The text between the <a> and the </a> is used as the caption for the link. It is common for the caption to be in blue underlined text. It is by the way a good idea to not have any blank spaces in the names of your HTML files, as this might create problems with some web servers. You can use an underscore, "_", to separate words in your file names.

To link to a page on another web site you need to give the full web address (the URL). For instance, to link to "Google" you need to write:

A link to **<a href="http://www.google.com">Google</a>.**

If you want the user's browser to open a new window for the linked page, (that way the user finds back to your page as soon as he or she closes the new window), use the attribute target:

A link to **<a href=http://www.google.com target='_blank'>Google</a>.**

5.3.2 External Document References

Syntax

```
<A HREF = "fname.htm"> Hyperlink CONTENT </A>
```

Example: <A HREF ="mypage.htm"> Visit to my main page </A>

Here Visit to my main page will be the name of link which will link to another document, mypage.html, Which is present in the current working directory. If the file is not present in the current directory, a relative or absolute path can be specified.

To move to a specific location on a Web page, *named anchors* can be used which locates the link to a specific point on a page.

It can be done in *two* steps :

Step 1: Point the location to be moved i.e., identify the location in a Web page by specifying the location name. This is done by using the NAME attribute of the <A> tag.

Syntax

```
<A NAME = "file name.html">
```

Example: <A NAME = "loc.html ">

It specifies that location to be moved to loc.html.

Step 2: While moving to a particular Web page and location on the Web page, in addition to the name of the Web page to be moved to, the name of location of the target on the Web page is needed.

Syntax

```
<A HREF = "file name.htm # location_name"> ... </A>
```

Notes

Example: <A HREF ="home.html # location "> come to my page </A>

5.3.3 Internal Document References

Sometimes, a move has to be targeted to a particular location, the above two steps need to be performed as before i.e., identify a location with a name and then jump to that location using the name. The only difference is that the *filename.* htm now will be the current *filename.htm.*

Syntax

```
<A NAME = "location name">
<A HREF = "#location name"> Text </A>
```

The absence of the *filename. htm* before the # symbol indicates that jump required within the same document.

Example: <A NAME = "location">

```
<A HREF = "#location"> come to my page </A>
```

Come to my page becomes a Hotspot and leads to a location named point 1 in the same document.

Everything is aligned to the left side of each cell. This is the default setting.

5.3.4 Creating Link Lists

If you want to preserve all of your links and space of your blog you can use **scrollable** box which contains link list.

Caution It is not nice when list of links is too long because it occupies too much space on your blog or web page.

To create scrollable list of links like this you should add this code:

```
<div style="overflow:auto;width:100%;height:150px;border:1px solid #ccc;">
<a href="http://interestingwebs.blogspot.com/2008/12/advices-to-increase-
blog-traffic-and.html">Advices to increase blog traffic</a>
<br>
<a href="http://interestingwebs.blogspot.com/2008/12/easy-way-to-put-code-
snippets-in-blog.html">How to display HTML code in blog or web</a>
<br>
<a href="http://interestingwebs.blogspot.com/2008/12/how-to-add-falling-
snow-effect-to
blog.html">How to add a falling snow effect to blog (blogger or blogspot)</
a>
<br>
<a href="http://interestingwebs.blogspot.com/2008/12/improve-blog-
experience-with addthis.html">Improve blog experience with bookmarking and
sharing service (AddThis
button)</a>
<br>
<a href="http://interestingwebs.blogspot.com/2008/11/adding-categories-to-
blogspot-blogger.html">Adding categories to blogspot (blogger)</a>
</div>
```

Of course in <a> tags you should put yours links. And you can add as many links as you want. The <div> tag defines a division or a section in an HTML document.

Did u know? In div element overflow property is set to auto which mean that if the content is clipped, the browser should display a scroll-bar to see the rest of the content.

5.3.5 Inserting Inline Images

<Img> causes an "inline image" to be inserted into the output. The image will be retrieved and rendered as if it were just another part of the text. Inline images can occur within headings, or paragraphs, almost anywhere in fact, except body (in other words, you can have a 'free floating' <img> tag—it must be contained within some other element.)

Like <hr>, this is an empty element. That is, there must be *no end-tag*. You will sometimes see <hr> used with an end-tag (e.g as a container around a caption), but this is obsolete usage.

<Img> has 1 required attributes src as well as 3 optional attributes.

src

The src attribute is used to specify the URL of the image (i.e. the address or filename the browser uses to retrieve the image file), e.g.

```
<img src="http://planet-earth.bogus.us/icons/secret.pictures.gif">
<img src="grandchild.gif">
```

alt

alt is used to provide an text alternative to the image for readers whose browsers do not support graphics (or for visually impaired readers using alternative display devices). Although not required, the use of the alt attribute is nearly always appropriate and is strongly recommended. The only exception might be cases where the image is strictly decorative or of generic character. In this case the default text chosen by the browser (typically something like "[IMAGE]" may be adequate.

align

align can take one of three values:

- top
- middle
- bottom

and is used to indicate how the browser should align the image with the adjacent text.

bottom: align the bottom of the image with the bottom of text

middle: align the middle of the image with the middle of text

top: align the top of the image with the top of text.

Images can also be retrieved using by means of a hypertext link using the <a> tag. The key difference between an inline image and an image retrieved with the <a> tag is that an inline image requires no action on the part of the reader; it is retrieved with the page just as if it were part of the text. With the <a> tag, the reader has to make a special action (e.g. clicking on a button) to retrieve the image.

Notes

Task Give examples of the attributes used with <img> tag.

5.3.6 Creating Image Links

Image links are constructed as you might expect, by embedding an <img> tag inside of an anchor element <a>. Like HTML text links, image links require opening and closing anchor tags, but instead of placing text between these opening and closing tags, the developer needs to place an image tag — with a valid source attribute value of course.

Example:

```
<!DOCTYPE html>
<html>
<body>
<p>Create a link of an image:
<a href="default.html">
<img src="smiley.gif" alt="HTML tutorial" width="32" height="32"></a></p>
<p>No border around the image, but still a link:
<a href="default.html">
<img border="0" src="smiley.gif" alt="HTML tutorial" width="32"
height="32"></a></p>
</body>
</html>
```

Output

Create a link of an image:

No border around the image, but still a link:

5.3.7 Image Maps

Image maps are images with clickable areas (sometimes referred to as "hotspots") that usually link to another page. If used tastefully, image maps can be really effective. If not, they can potentially confuse users.

To create an image map:

1. You need an image. Create an image using the usual method (i.e. via an image editor, then save as a gif or jpeg into your website's image folder).
2. Use the HTML <map> tag to create a map with a name. Inside this tag, you will specify where the clickable areas are with the HTML <area> tag
3. Use the HTML <img> tag to link to this image. In the img tag, use with the usemap attribute to define which image map to use (the one we just specified).

Example:

```
<img src ="/pix/mueller_hut/mueller_hut_t.jpg"
```

Notes

```
width="225" height="151" border="0"
alt="Mueller Hut, Mount Cook, and I"
usemap ="#muellermap" />
<map id ="muellermap"
name="muellermap">
 <area shape ="rect" coords ="90,80,120,151"
 href ="javascript:alert('Me');" target="_blank" alt="Me" />
 <area shape ="poly" coords ="55,55,120,80,90,80,90,100,70,100,20,80,55,55"
 href ="http://en.wikipedia.org/wiki/Mount_Cook" target="_blank" alt="Mount
Cook" />
 <area shape ="poly" coords ="145,80,145,100,215,90,215,80,180,60,145,80"
 href ="http://www.natural-environment.com/places/mueller_hut.php"
target="_blank" alt="Mueller Hut" />
</map>
```

In our example, we use the <area> tag in conjunction with the shape and coord attributes. These accept the following attributes:

shape	Defines a shape for the clickable area. Possible values: • default • rect • circle • poly
coords	Specifies the coordinates of the clickable area. Coordinates are specified as follows: • rect: left, top, right, bottom • circle: center-x, center-y, radius • poly: x1, y1, x2, y2, ...

5.3.8 Sending e-mail to Specific Link Address

Having a link that allows visitors to send email from your website can be a great addition to your site, making it easy for your visitors to send questions or comments. There is a special link for this action.

Email links are done much the same as links to other pages, using the <a href> tag.

An email link would require the following code:

```
<a href="mailto:youremailaddress">Email Me</a>
```

This will result in the visitor's email program opening a new email with your address already in the To: field.

If you wish to have a specific subject in the email, you can add it to the html code using subject= setting :

```
<a href="mailto:email@echoecho.com?subject=SweetWords"> Send Email</a>
```

Suppose you want an email link for your visitors containing specific text in the body of their message, simply add &body=:

<a href="mailto:email@echoecho.com?body=Please send me a copy of your new program!">Send Email</a>

Or combine all the options and allow your visitor to send email with the address, subject and text already entered.

```
        <a href=”mailto:email@echoecho.com?subject=SweetWords&body=Please send
me a copy of your new program!”>Email Me</a>
```

Frames divide a browser window into two or more document windows, each displaying a different document, or a different part of the same document.

Caution Frames in an HTML document can cause a web page to appear to be divided into multiple, scrollable regions.

For each frame, you can assign a name, a source document locator, dimensions, border alignment and decorations, scroll and resize behaviors, loading and unloading behavior, file and topic maps, and style sheets.

- ***Names:*** You can place an anchor in any frame, link to any addressable object, and place the object into any named frame.
- ***Source document locator:*** You can use whatever addressing schemes your user agent supports, including URLs and filenames.
- ***Dimensions:*** You can rigidly or flexibly layout a two-dimensional grid of rectangular blocks.
- ***Border alignment and decoration:*** You can adjust the position of the left and right margins, the top and bottom margins, and the alignment of the frame. You can also make the borders of a frame invisible.
- ***Scrolling:*** Frames can have scrollbars, no scrollbars, or you can let the browser turn them on if the document is larger than the current horizontal or vertical size of the frame.
- ***Resizable:*** Frames are normally resizable in the browser, but that can be disabled so the frame may not be resized at the user agent.
- ***Loading and unloading:*** You can provide a script to be run when the user agent finishes loading all frames or when the user exit the document.
- ***File and topic maps:*** You can place a file or topic navigator into a frame. The navigator might be a collapsible listing of file system, a listing of document headings, thumbnails of images in a document, or an index of any element type.

These properties make possible:

- ***Static frames:*** Elements that a user should always see, such as button bars, logos, copyright notices, and title graphics, can be placed in individual frames that are locked into place on the user agent window.
- ***Live frames:*** Documents, icons, interactive forms, videos, multimedia, topic maps, and anything else that can react to user input or programmed activity.
- As a user navigates a site in “live” frames, the contents of static frames remain fixed, even though adjoining frames redraw.
- ***Functional tables of contents:*** A frame can contain interactive tables of contents (TOCs) with links that, when clicked, display results in an adjoining frame. These TOCs can be static or interactive with collapsible lists, graphical maps of document structure, or displays of file and link architectures.

Notes

- ***Single-page query and answer displays:*** Frames designed side-by-side permit queries to be posed and answered on the same page, with one frame holding the query form, and the other presenting the results.

Syntax

```
<HTML>
<HEAD>
</HEAD>
<FRAMESET rows="33%,33%,33%">
  <FRAMESET cols="50%,50%">
  <FRAME name="frame1">
  <FRAME name="frame2">
  </FRAMESET>
  <FRAME name="frame3">
  <FRAME name="frame4">
</FRAMESET>
<BODY>
```

...contents to display in non-frame-capable user agent...

```
</BODY>
</HTML>
```

Try to remember the following basic frameset concepts:

- The opening FRAMESET tag defines the actual size and layout of the frames.
- Using the ROWS and COLS attributes of the FRAMESET tag, you define frames, then **fill** them with the FRAME element, or **subdivide** them with nested FRAMESET elements.
- The number of entries in ROWS or COLS attribute values determine how many frames are displayed and what size they are. If you have both ROWS and COLS, multiply the number of entries in each attribute value to calculate the total number of frames (i.e., three entries in the ROWS attribute and four entries in the COLS attribute create a grid of twelve frames).
- The FRAME elements fill the frames defined by FRAMESET, and are applied in the frame set in a left-to-right, top-to-bottom order.
- FRAME elements which include the SRC attribute, define the contents of each frame you create.
- Frames are **always** rectangular.

The FRAMESET Element

A frameset divides the browser window into rectangular regions. Each such region can be:

- A **frame**, which displays one document. A frame is represented by a FRAME element.
- Another frame set, which is itself further divided into frames.

Example: A frame set can contain a frame, plus another frame set containing two frames, resulting in three frames in all.

An HTML document that contains frames basically replaces the BODY element with the FRAMESET element.

Notes

- ***The rows and cols attributes:*** The attributes of the FRAMESET element are COLS (columns) and ROWS. They determine how many frames the frame set is divided into. These attributes may be blank, or may consist of a list of one or more values separated by commas or spaces. Each such value determines the width (for columns) and height (for rows) of the regions; the number of width and height values supplied determines how many rows and columns, respectively, are created. The default for each is one.

  For example if you have:

  <FRAMESET cols="20%,30%,50%">

  in which there is no ROWS value, the frame set is divided vertically into three regions: the first region's width is 20% of the current frame set (or browser window if this frame set is at the top level), the second region's width is 30%, and the third region's width is 50%. When there is only one frame set in the document, these widths apply to the entire browser window. Similarly, when there is a ROWS value but no COLS value, the frame set is divided horizontally into regions. When values are supplied for both attributes, the frame set is divided into a grid of rows and columns.

  The ROWS and COLS attributes take comma-separated lists of values. These values can be absolute pixel values, percentage values between 1 and 100, or relative scaling values. The number of rows and columns is implicit in the number of values in the respective list. Since the total height of all the rows must equal the height of the window, row heights might be normalized to achieve this.

Notes If the rows (or cols) attribute values are unspecified, then the number of rows (or columns) is assumed to be one, and it may be arbitrarily sized to fit.

- *Syntax of Value List*

  value

  A simple numeric value is assumed to be a fixed size in pixels. The result of this value varies with the size of a viewer's window. Fixed pixel values are usually used with one or more of the relative size values described below. You might use a fixed value if you want a graphic, such as an image map, to fill an entire frame and you want to ensure that the frame is big enough to display the entire image. User agents can be expected to override a specified pixel value to ensure that the total proportions of a frame are 100% of the width and height of a user's window.

  value%

  This is a simple percentage value between 1 and 100. If the total is greater than 100, all percentages are scaled down. If the total is less than 100, and relative-sized frames exist, extra space is given to them. If there are no relative-sized frames, all percentages are scaled up to match a total of 100%.

  For example, suppose you assign the ROWS attribute a value of "50%,50%,50%". Each entry is 50%, which is one third of the sum of all the entries (150%), so the browser assigns the frame sizes proportionately, giving each frame one third of the browser height.

  value*

  The value on this field is optional. A single '*' character is a "relative-sized" frame and is interpreted as a request to give the frame all remaining space. If multiple relative-sized frames are specified, the remaining space is divided evenly among them. If there is a value in front of the '*', that frame gets that much more relative space. "2*,*" would give 2/3 of the space to the first frame, and 1/3 to the second.

Example: Setting Frame width and height

There are three ways to specify the width or height of a frame:

- As a percentage of the area allotted to the parent frame set.
- As an absolute (specific) number of screen pixels (e.g., 250).
- As a 'relative size'.

Example for 3 rows, the first and the last being smaller than the center row:

```
<FRAMESET rows="20%,60%,20%">
```

Example for 3 rows, the first and the last being fixed height, with the remaining space assigned to the middle row:

```
<FRAMESET rows="100,*,100">
```

A 'relative size' is specified with an asterisk, e.g., '1*', '2*', '3*' ('1*' can also be written simply as '*'). This is interpreted as follows: after all widths (or heights) specified as percentages or absolute amounts have been allocated to the corresponding frames, the remaining space is allocated to frames whose widths (or heights) have been specified as a relative size. The amount of space allocated to a frame is proportional to the number in front of the asterisk.

Example:

```
<FRAMESET rows="30%,400,*,2*">
       <FRAME ... >
       <FRAME ... >
       <FRAME ... >
       <FRAME ... >
</FRAMESET>
```

Suppose the browser window is currently 1000 pixels high. The first frame gets 30% of the total height, that is, 300 pixels; the second frame gets 400 pixels, since an absolute amount was specified. This leaves 300 pixels to be divided between the other two frames. The fourth frame's height is specified as '2*', so it is twice as high as the third frame, whose height is only '*' (1*). Therefore the third frame is 100 pixels high and the fourth is 200 pixels high.

The FRAME Element

The FRAME element defines a single frame in a frameset. It has 7 possible attributes: SRC, NAME, FRAMEBORDER, MARGINWIDTH, MARGINHEIGHT, SCROLLING, and NORESIZE. The FRAME tag is not a container so it has no matching end tag.

The attributes of the FRAME element type are as follows:

`src="address"`

Specifies the address of the document to be displayed in the frame. When omitted the frame is displayed as a blank space.

`name="window_name"`

The NAME attribute is used to assign a name to a frame so it can be targeted by links in other documents (These are usually from other frames in the same document.) The NAME attribute is optional; by default all windows are unnamed. Names must begin with an alphabetic character.

Named frames can have their window contents targeted with the TARGET attribute.

```
frameborder="1|0"
```

FRAMEBORDER is a boolean which triggers the display of frame separators around the frame. When set to "1" a separator is drawn on every side next to another frame. When set to "0" the decision to draw separators is left to surrounding frames. Indeed in this case separators might still be drawn because FRAMEBORDER is set "1" on adjacent frames. Default is "1".

```
marginwidth="value"
```

The MARGINWIDTH attribute is used when the document author wants some control of the margins for this frame. If specified, the value for MARGINWIDTH is in pixels. Margins can not be less than one; this insures that frame objects won't touch frame edges, and can't be specified in a way that leaves no space for the document contents. The MARGINWIDTH attribute is optional; by default, all frames default to letting the browser decide on an appropriate margin width.

```
marginheight="value"
```

The MARGINHEIGHT attribute is just like MARGINWIDTH above, except it controls the upper and lower margins instead of the left and right margins.

noresize

The NORESIZE attribute doesn't require a value. It can simply be used as a flag to indicates that the frame is not resizable by the user. Users typically resize frames by dragging a frame edge to a new position. Note that if any frame adjacent to an edge is not resizable, that entire edge is restricted from moving. This affects the resizability of other frames. The NORESIZE attribute is optional; by default all frames are resizable.

scrolling="yes|no|auto"

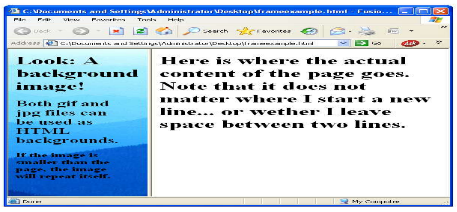
The SCROLLING attribute indicates whether the frame should have scrollbars or not. "yes" results in scrollbars always being visible on that frame. "no" results in scrollbars never being visible. "auto" instructs the browser to decide whether scrollbars are needed, and to place them where necessary. That is, with "auto", the frame has scrollbars only if the document is larger than the current size of the frame.

Did u know? The SCROLLING attribute is optional; the default value is "auto."

Example:

```
<frameset cols="25%,75%">
  <frame src="frame_a.htm">
  <frame src="frame_b.htm">
</frameset>
```

The output will be:


C:\Documents and Settings\Administrator\Desktop\frameexample.html - Fusio...
File Edit View Favorites Tools Help
Back Search Favorites
Address C:\Documents and Settings\Administrator\Desktop\frameexample.html
Go
Done
My Computer

Look: A background image!

Both gif and jpg files can be used as HTML backgrounds.

If the image is smaller than the page, the image will repeat itself.

Here is where the actual content of the page goes. Note that it does not matter where I start a new line... or wether I leave space between two lines.

Self Assessment

Fill in the blanks:

8. Links are defined with the .... tag.
9. The text between the <a> and the </a> is used as the .... for the link.
10. You can use an .... to separate words in your file names.
11. If the file is not present in the current directory, a .... path can be specified.
12. If you want to preserve all of your links and space of your blog you can use .... box which contains link list.
13. The .... tag defines a division or a section in an HTML document. In div element overflow property is set to auto.
14. <Img> causes an ".........................." to be inserted into the output.
15. Image .... are constructed by embedding an <img> tag inside of an anchor element <a>.

5.4 Summary

- Tables are defined with the <table> tag.
- Constructing an HTML table consists of describing the table between the beginning table tag, <TABLE>, and the ending table tag, </TABLE>.
- Links are the essence of HTML — they are what makes it unique. Links are defined with the <a> tag.
- If you do not specify a border attribute the table will be displayed without any borders.
- <Img> causes an "inline image" to be inserted into the output.
- Image maps are images with clickable areas (sometimes referred to as "hotspots") that usually link to another page.
- Email links are done much the same as links to other pages, using the <a href> tag.
- The key difference between an inline image and an image retrieved with the <a> tag is that an inline image requires no action on the part of the reader.

5.5 Keywords

alt: alt is used to provide an text alternative to the image for readers whose browsers do not support graphics.

Cell tags: <TD> </TD>

Image links: Image links are constructed as you might expect, by embedding an <img> tag inside of an anchor element <a>.

Links: Links are the essence of HTML — they are what makes it unique

Row tags: <TR> </TR>

src: The src attribute is used to specify the URL of the image.

Table tags: <TABLE> </TABLE>

Table: Tables are defined with the <table> tag.

5.6 Review Questions

1. Explain how HTML Tables are created?
2. Define and Explain the steps involved in creating HTML Tables.
3. Discuss in brief about Linking Document.
4. What are the three attributes that can be specified with the <BODY> tag? Explain each of them.
5. Define and explain Hyperlinks and their types.
6. Justify the use of link list creation in html document.
7. Explain how to display a table with borders.
8. Discuss how to link to a page on another web site. Illustrate with example.
9. Explain the steps used to move to a specific location on a Web page.

Answers: Self Assessment

1.	<table>	2.	Td
3.	data cell	4.	<tr>
5.	</TR>.	6.	Border
7.	<th>	8.	<a>
9.	caption	10.	underscore, "_"
11.	relative or absolute	12.	Scrollable
13.	<div>	14.	inline image
15.	links		

5.7 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Notes

Online links

http://www.temple.edu/cs/web/tables.html

http://www.w3schools.com/html/tryit.asp?filename=tryhtml_imglink

http://www.tizag.com/htmlT/htmlimagelinks.php

http://www.quackit.com/html/tutorial/html_image_maps.cfm

http://www.echoecho.com/htmllinks11.htm

http://interestingwebs.blogspot.in/2008/12/how-to-create-scrollable-link-list.html

http://www-sul.stanford.edu/tools/tutorials/html2.0/img.html

Unit 6: Introduction to JavaScript

CONTENTS

Objectives

After studying this unit, you will be able to:

- Explain the Overview of JavaScript
- Describe how to tag JavaScript with HTML
- Discuss about the Browsers compatibility with JavaScript
- Describe the Data types in JavaScript
- Analyse the typecasting in JavaScript
- Explain the Arrays in JavaScript

Introduction

Java script is a browser-interpreted language that was created to access all elements of HTML and the browser. The processing is done entirely by the client-side browser which makes it very useful tool to handle processing which would have otherwise been checked server-side, thereby reducing overhead. Java script is also used to increase user interaction, animate objects, create drop down navigation, grab data from databases, and more!

JavaScript is most commonly used as a client side scripting language. This means that JavaScript code is written into an HTML page. When a user requests an HTML page with JavaScript in it, the script is sent to the browser and it's up to the browser to do something with it. The fact that the script is in the HTML page means that your scripts can be seen and copied by whoever views your page. Nonetheless, to my mind this openness is a great advantage, because the flip side is that you can view, study and use any JavaScript you encounter on the WWW.

JavaScript can be used in other contexts than a Web browser. Netscape created server-side JavaScript as a CGI-language that can do roughly the same as Perl or ASP.

JavaScript is not a programming language in strict sense. Instead, it is a scripting language because it uses the browser to do the dirty work. If you command an image to be replaced by another one, JavaScript tells the browser to go do it. Because the browser actually does the work, you only need to pull some strings by writing some relatively easy lines of code. That's what makes JavaScript an easy language to start with.

But don't be fooled by some beginner's luck: JavaScript can be pretty difficult, too. First of all, despite its simple appearance it is a full fledged programming language: it is possible to write quite complex programs in JavaScript. This is rarely necessary when dealing with web pages, but it is possible. This means that there are some complex programming structures that you'll only understand after protracted studies.

Secondly, and more importantly, there are the browser differences. Though modern web browsers all support JavaScript, there is no sacred law that says they should support exactly the same JavaScript. A large part of this site is devoted to exploring and explaining these browser differences and finding ways to cope with them.

So basic JavaScript is easy to learn, but when you start writing advanced scripts browser differences (and occasionally syntactic problems) will creep up.

6.1 Origin of JavaScript

JavaScript was originally developed in Netscape, by Brendan Eich. Battling with Microsoft over the Internet, Netscape considered their client-server solution as a distributed OS, running a portable version of Sun Microsystems' Java. Because Java was a competitor of C++ and aimed at professional programmers, Netscape also wanted a lightweight interpreted language that would complement Java by appealing to nonprofessional programmers, like Microsoft's Visual Basic.

Developed under the name *Mocha*, *LiveScript* was the official name for the language when it first shipped in beta releases of Netscape Navigator 2.0 in September 1995, but it was renamed JavaScript when it was deployed in the Netscape browser version 2.0B3.

The change of name from LiveScript to JavaScript roughly coincided with Netscape adding support for Java technology in its Netscape Navigator web browser. The final choice of name caused confusion, giving the impression that the language was a spin-off of the Java programming language, and the choice has been characterized by many as a marketing ploy by Netscape to give JavaScript the cachet of what was then the hot new web programming language.

Self Assessment

Fill in the blanks:

1. Developed under the name ...., LiveScript was the official name for the language when it first shipped in beta releases of Netscape Navigator 2.0.
2. The change of name from .... to JavaScript roughly coincided with Netscape adding support for Java technology in its Netscape Navigator web browser.

6.2 JavaScript and Web

JavaScript is a scripting language designed primarily for adding interactivity to Web pages and creating Web applications.

JavaScript is the scripting language implemented on web which is used to add several functionalities, validating the form, as well as communicating with servers etc.

Example:

```
<!DOCTYPE html>
<html>
<head>
<script>
function displayDate()
{
document.getElementById("demo").innerHTML=Date();
}
</script>
</head>
<body>

<h1>WELCOME TO JAVA SCRIPT</h1>
<p id="demo">to get current date click on display date.</p>

<button type="button" onclick="displayDate()">Display Date</button>

</body>
</html>
```

Self Assessment

Fill in the blanks:

3. .... is a scripting language designed primarily for adding interactivity to Web pages and creating Web applications.
4. JavaScript is the scripting language implemented on ....

6.3 JavaScript

JavaScript is a programming language used to make web pages interactive. It runs on your visitor's computer and doesn't require constant downloads from your website. JavaScript is often used to create polls and quizzes.

6.3.1 Java and JavaScript

JavaScript is *not* the same as Java. Although the names are much alike, JavaScript is primarily a scripting language for use within HTML pages, while Java is a real programming language that does quite different things from JavaScript. In addition Java is much harder to learn. It was developed by Sun for use in pretty much anything that needs some computing power.

JavaScript was developed by Brendan Eich, then working at Netscape, as a client side scripting language (even though there's no fundamental reason why it can't be used in a server side environment).

Originally the language was called Live Script, but when it was about to be released Java had become immensely popular (and slightly hypey). At the last possible moment Netscape changed the name of its scripting language to "JavaScript". This was done purely for marketing reasons. Worse, Eich was ordered to "make it look like Java". This has given rise to the idea that JavaScript is a "dumbed-down" version of Java. Unfortunately there's not the slightest shred of truth in this story.

Notes Java and JavaScript both descend from C and C++, but the languages (or rather, their ancestors) have gone in quite different directions. You can see them as distantly related cousins. Both are object oriented (though this is less important in JavaScript than in many other languages) and they share some syntax, but the differences are more important than the similarities.

Task Analyse various applications of javascript.

6.3.2 What can JavaScript do?

- ***JavaScript gives HTML designers a programming tool:*** HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax! Almost anyone can put small "snippets" of code into their HTML pages.
- ***JavaScript can put dynamic text into an HTML page:*** A JavaScript statement like this: document. write ("<h1>" + name + "</h1>") can write a variable text into an HTML page.
- ***JavaScript can react to events:*** A JavaScript can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element.
- ***JavaScript can read and write HTML elements:*** A JavaScript can read and change the content of an HTML element.
- ***JavaScript can be used to validate data:*** A JavaScript can be used to validate form data before it is submitted to a server. This saves the server from extra processing.
- ***JavaScript can be used to detect the visitor's browser:*** A JavaScript can be used to detect the visitor's browser, and - depending on the browser - load another page specifically designed for that browser.
- ***JavaScript can be used to create cookies:*** A JavaScript can be used to store and retrieve information on the visitor's computer.

Self Assessment

Fill in the blanks:

5. JavaScript can put .... text into an HTML page.

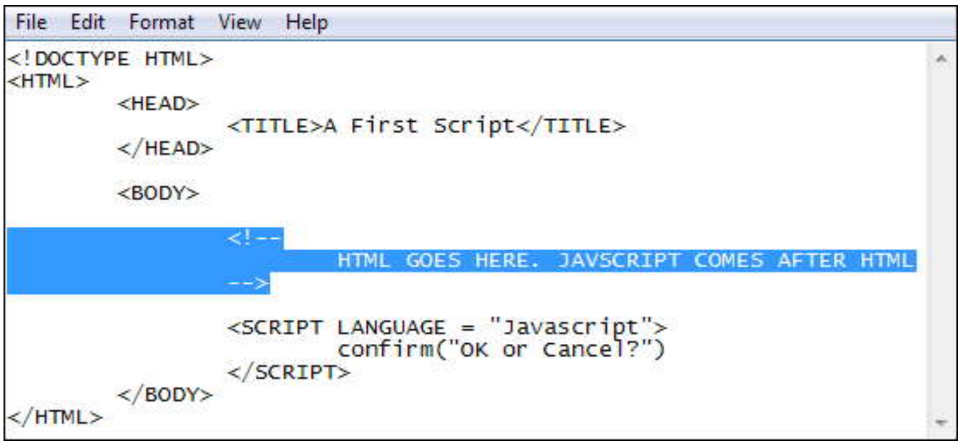
6. A JavaScript can be used to validate form data before it is submitted to a ....

6.4 Script (<script>)Tag

In Javascript, SCRIPT tags can be inserted into three places: in between the two BODY tags, in between the two HEAD tags, and as a link to an external file, also in the HEAD section.

Some advocate putting the SCRIPT tags in the BODY section, right at the end. Like this:

Figure 6.1: SCRIPT Tags in the BODY Section

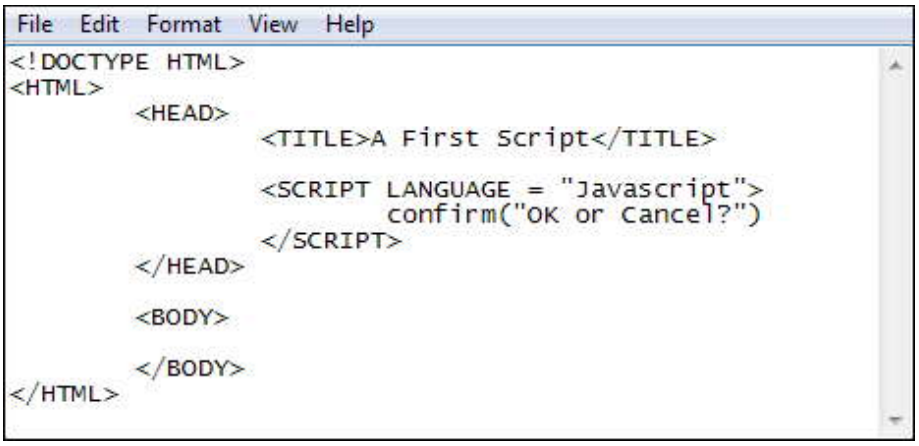


```
<!DOCTYPE HTML>
<HTML>
        <HEAD>
                <TITLE>A First Script</TITLE>
        </HEAD>

        <BODY>

                <!--
                        HTML GOES HERE. JAVSCRIPT COMES AFTER HTML
                -->

                <SCRIPT LANGUAGE = "Javascript">
                        confirm("OK or Cancel?")
                </SCRIPT>
        </BODY>
</HTML>
```

The reason to do it this way is because the web page itself will have loaded before the script is read. If it's in the HEAD section, the script will be parsed before any HTML or CSS elements are loaded. If your Javascript references any of the web page's elements, there may be a slight delay in the fancy effects you want to apply, or it may just not work at all.

For the most part, though, we'll place our script tags in the HEAD section of the HTML. Like this:

Figure 6.2: Script Tags in the HEAD Section



```
<!DOCTYPE HTML>
<HTML>
        <HEAD>
                <TITLE>A First Script</TITLE>

                <SCRIPT LANGUAGE = "Javascript">
                        confirm("OK or Cancel?")
                </SCRIPT>
        </HEAD>

        <BODY>

        </BODY>
</HTML>
```

Notes

Self Assessment

Fill in the blanks:

7. In Javascript, ... tags can be inserted into three places: in between the two BODY tags, in between the two HEAD tags, and as a link to an external file, also in the HEAD section.

8. If your Javascript references any of the web page's ...., there may be a slight delay in the fancy effects you want to apply, or it may just not work at all.

6.5 Browsers Compatibility

Browsers that do not support JavaScript, will display JavaScript as page content. To prevent them from doing this, and as a part of the JavaScript standard, the HTML comment tag should be used to "hide" the JavaScript.

Just add an HTML comment tag <!— before the first JavaScript statement, and a —> (end of comment) after the last JavaScript statement, like this:

```
<html>
<body>
<script type="text/JavaScript">
<!—
document.getElementById("demo").innerHTML=Date();//—>
</script>
</body>
</html>
```

The two forward slashes at the end of comment line (//) is the JavaScript comment symbol. This prevents JavaScript from executing the —> tag.

Self Assessment

Fill in the blanks:

9. Browsers that do not support JavaScript, will display JavaScript as ....

10. Add an HTML comment tag .... before the first JavaScript statement, and a —> (end of comment) after the last JavaScript statement.

6.6 Data Types

A program can do many things, including calculations, sorting names, preparing phone lists, displaying images, validating forms, ad infinitum. But in order to do anything, the program works with the data that is given to it.

Data types specify what kind of data, such as numbers and characters, can be stored and manipulated within a program. JavaScript supports a number of fundamental data types.

Did u know? These types can be broken down into two categories, primitive data types and composite data types.

6.6.1 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
- ***Numeric Literals:*** JavaScript supports both integers and floating-point numbers. Integers are whole numbers and do not contain a decimal point; e.g., 123 and –6. Integers can be expressed in decimal (base 10), octal (base 8), and hexadecimal (base 16), and are either positive or negative values.

  Floating-point numbers are fractional numbers such as 123.56 or –2.5. They must contain a decimal point or an exponent specifier, such as 1.3e–2. The letter "*e*" for exponent notation can be either uppercase or lowercase.

  JavaScript numbers can be very large (e.g., 10^{308} or 10^{-308}).

Table 6.1: Numeric Literals

12345	Integer
23.45	Float
.234E–2	scientific notation
.234e+3	scientific notation
0x456fff	Hexadecimal
0x456FFF	Hexadecimal
0777	Octal

- ***String Literals and Quoting:*** String literals are rows of characters enclosed in either double or single quotes. The quotes must be matched. If the string starts with a single quote, it must end with a matching single quote, and likewise if it starts with a double quote, it must end with a double quote. Single quotes can hide double quotes, and double quotes can hide single quotes:

  "This is a string"

  'This is another string'

  "This is also 'a string' "

  'This is "a string"'

An empty set of quotes is called the null string. If a number is enclosed in quotes, it is considered a string; e.g., *"5"* is a string, whereas *5* is a number.

Notes

Strings are called constants or literals. The string value *"hello"* is called a string constant or literal. To change a string requires replacing it with another string.

Strings can contain escape sequences (a single character preceded with a backslash), as shown in Table 6.2. Escape sequences are a mechanism for quoting a single character.

Table 6.2: Escape Sequences

Escape Sequence	What It Represents
\'	Single quotation mark
\"	Double quotation mark
t	Tab
n	Newline
r	Return
f	Form feed
b	Backspace
e	Escape
\\	Backslash
Special Escape Sequences	
XXX	The character with the Latin-1 encoding specified by up to three octal digits XXX between 0 and 377. \251 is the octal sequence for the copyright symbol.
xXX	The character with the Latin-1 encoding specified by the two hexadecimal digits XX between 00 and FF. \xA9 is the hexadecimal sequence for the copyright symbol.
uXXXX	The Unicode character specified by the four hexadecimal digits XXXX. \u00A9 is the Unicode sequence for the copyright symbol.

Example: Illustrating strings in java script

```
<html>
<head>
<body>
<pre>
<font size="+2">
<script language="JavaScript">
<!— Hide script from old browsers.
document.write("\t\tHello\nworld!\n");
document.writeln("\"Nice day, Mate.\"\n");
document.writeln('Smiley face:<font size="+3"> \u263a\n');
//End hiding here. —>
</script>
</pre>
</body>
</html>
```

Output

```
Hello
```

```
world!
"Nice day, Mate."

Smiley face: :&
```

Explanation

1. The escape sequences will work only if in a *<pre>* tag or an alert dialog box.
2. The JavaScript program starts here.
3. The *write()* method sends to the browser a string containing two tabs (*t**t*), *Hello*, a newline (*n*), world!, and another newline (*n*).
4. The *writeln()* method sends to the browser a string containing a double quote (\"), *Nice day, Mate.*, another double quote (\"), and a newline (*n*). Since the *writeln()* method automatically creates a newline, the output will display two newlines: the default value and the *n* in the string.
5. This string contains a backslash sequence that will be translated into Unicode. The Unicode hexidecimal character 233a is preceded by a *u*.

The process of joining strings together is called concatenation. The string concatenation operator is a plus sign (+). Its operands are two strings. If one string is a number and the other is a string, JavaScript will still concatenate them as strings.

Caution If both operands are numbers, the + will be the addition operator.

The following examples output *"popcorn"* and *"Route 66"*, respectively.

```
document.write("pop" + "corn");
document.write("Route " + 66);
```

The expression *5 + 100* results in *105*, whereas *"5" + 100* results in *"5100"*.

Boolean Literals

Boolean literals are logical values that have only one of two values, *true* or *false*. You can think of the values as yes or no, on or off, or 1 or 0. They are used to test whether a condition is true or false.

Did u know? When using numeric comparison and equality operators, the value *true* evaluates to 1 and *false* evaluates to 0.

answer1 = true;

or

if (answer2 == false) { do something; }

6.6.2 Composite Data Types

We mentioned that there are two types of data: primitive and composite. The primitive types: numbers, strings, and Booleans—each storing a single value. Composite data types, also called complex types, consist of more than one component.

Notes

- Objects contain properties and methods;
- Arrays contain a sequential list of elements;

Functions contain a collection of statements.

Task Compare and contrast primitive data types and composite data types.

Self Assessment

Fill in the blanks:

11. .... data types are the types that can be assigned a single literal value.
12. .... are rows of characters enclosed in either double or single quotes.

6.7 Creating Values

Here, we will demonstrate the use of JavaScript for accessing the values of form elements.

6.7.1 Value for Text Input Element

Figure 6.3: Text Input Element

To obtain a reference to a text input element, here is some sample code:

```
oText = oForm.elements["text_element_name"]; OR
oText = oForm.elements[index];
```

In the code above, "index" is the position of the element in the 0-based elements array, and oForm is the form object reference obtained using the document.forms collection:

```
oForm = document.forms[index];
```

To get the value of the text input element, we can use the value property of the text input object:

```
text_val = oText.value;
```

As an example, if we have the following text input element:

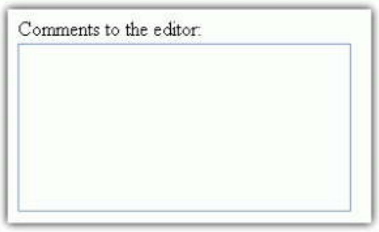
```
<input type="text" name="name" id="txt_name" size="30" maxlength="70">
```

We can access the value of the element like this:

```
name = oForm.elements["name"].value;
```

6.7.2 Value for Text Area Element

Figure 6.4: Text Area Element



The code for obtaining a reference to a textarea element is very similar:

```
oTextarea = oForm.elements["textarea_element_name"];
```

To get the value entered by the user in the textarea field:

```
textarea_val = oTextarea.value;
```

As an example, if we have a textarea element like this:

```
<textarea name="address" id="txta_address" rows="3" cols="35"></textarea>
```

We can access the value entered by the user in this way:

```
address = oForm.elements["address"].value;
```

6.7.3 Value for Hidden Element

The code for obtaining a reference to a hidden input element:

```
oHidden = oForm.elements["hidden_element_name"];
```

To get the value of this element:

```
hidden_val = oHidden.value;
```

As an example, if we have a hidden input element in the form defined like this:

```
<input type="hidden" name="number_of_skillsets" value="1">
```

We can get the hidden input element's value like this:

```
number_of_skillsets = oForm.elements["number_of_skillsets"].value;
```

Self Assessment

Fill in the blanks:

13. To get the value of the .... element, we can use the value property of the text input object
14. The code for obtaining a reference to a .... element is oTextarea = oForm.elements["textarea_element_name"].

6.8 Type Casting

It's also possible to convert values using a process called type casting. Type casting allows you to access a specific value as if it were of a different type. Three type casts are available in JavaScript:

- Boolean(value) – casts the given value as a Boolean
- Number(value) – casts the given value as a number (either integer or floating-point)
- String(value) – casts the given value a string

Casting a value using one of these three functions creates a new value that is a direct conversion of the original. This can lead to some unexpected results. The Boolean() type cast returns true when the value is a string with at least one character, a number other than 0, or an object (discussed in the next section); it returns false when the value is an empty string, the number 0, undefined , or null . The following code snippet can be used to test type casting as a Boolean:

```
var b1 = Boolean(""); //false - empty string
var b2 = Boolean("hi"); //true - non-empty string
var b3 = Boolean(100); //true - non-zero number
var b4 = Boolean(null); //false - null
```

Notes

```
var b5 = Boolean(0); //false - zero
var b6 = Boolean(new Object()); //true – object
```

The Number() type cast works in a manner similar to parseInt() and parseFloat() , except that it converts the entire value, not just part of it. Remember that parseInt() and parseFloat() only convert up to the first invalid character (in strings), so "4.5.6" becomes "4.5" . Using the Number() type cast, "4.5.6" becomes NaN because the entire string value cannot be converted into a number. If a string value can be converted entirely, Number() decides whether to use parseInt() or parseFloat() . The following table illustrates what happens when Number() is used on various values:

Usage	Result
Number(true)	1
Usage Number(false)	0
Number(undefined)	NaN
Number(null)	0
Number("5.5")	5.5
Number("56")	56
Number("5.6.7")	NaN
Number(new Object())	NaN
Number(100)	100

The last type cast, String() , is the simplest because it can accurately convert any value to a string value. To execute the type cast, it simply calls the toString() method of the value that was passed in, which converts 1 to "1", true to "true", false to "false", and so on. The only difference between type casting as a string and using toString() is that the type cast can produce a string for a null or undefined value without error:

```
var s1 = String(null); //"null"
var oNull = null;
var s2 = oNull.toString(); //won't work, causes an error
```

Type casting is very helpful when dealing with the loosely typed nature of JavaScript, although youshould ensure that only proper values are used.

Self Assessment

Fill in the blanks:

15. .... allows you to access a specific value as if it were of a different type.

16. Type casts .... casts the given value as a number.

6.9 Arrays

An array is a special variable, which can hold more than one value, at a time.

If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

```
cars1="mercedes";
cars2="ferari";
cars3="BMW";
```

However, what if you want to loop through the cars and find a specific one? And what if you had not 3 cars, but 300?

The best solution here is to use an array!

Notes An array can hold all your variable values under a single name. And you can access the values by referring to the array name.

Each element in the array has its own ID so that it can be easily accessed.

6.9.1 Creating Arrays

An array can be defined in three ways.

The following code creates an Array object called myCars:

1:

```
var myCars=new Array(); // regular array (add an optional integer
myCars[0]="Mercedes"; // argument to control array's size)
m y C a r s [ 1 ] = " F e r a r i " ;
myCars[2]="BMW";
```

2:

```
var myCars=new Array("Mercedes"," Ferari ","BMW"); // condensed array
```

3:

```
var myCars=[" Mercedes "," Ferari ","BMW"]; // literal array
```

Caution If you specify numbers or true/false values inside the array then the variable type will be Number or Boolean, instead of String.

6.9.2 Access an Array

You can refer to a particular element in an array by referring to the name of the array and the index number. The index number starts at 0.

The following code line:

```
document.write(myCars[0]);
Will result in the following output:
Mercedes
```

6.9.3 Modify Values in an Array

To modify a value in an existing array, just add a new value to the array with a specified Index number:

```
myCars[0]="Opel";
Now, the following code line:
document.write(myCars[0]);
```

will result in the following output:

```
Opel
```

Notes

Table 6.3: Different Methods Used in Array

Method	Description
concat()	Joins two or more arrays, and returns a copy of the joined arrays
join()	Joins all elements of an array into a string
pop()	Removes the last element of an array, and returns that element
push()	Adds new elements to the end of an array, and returns the new length
reverse()	Reverses the order of the elements in an array
shift()	Removes the first element of an array, and returns that element
slice()	Selects a part of an array, and returns the new array
sort()	Sorts the elements of an array
splice()	Adds/Removes elements from an array
toString()	Converts an array to a string, and returns the result
unshift()	Adds new elements to the beginning of an array, and returns the new length
valueOf()	Returns the primitive value of an array

Self Assessment

Fill in the blanks:

17. An .... is a special variable, which can hold more than one value, at a time.

18. Each element in the array has its own .... so that it can be easily accessed.

6.10 Summary

- Java script is a browser-interpreted language that was created to access all elements of HTML and the browser.
- JavaScript is the scripting language implemented on web which is used to add several functionalities, validating the form, as well as communicating with servers etc.
- JavaScript is most commonly used as a client side scripting language.
- ***JavaScript is a scripting language*** designed primarily for adding interactivity to Web pages and creating Web applications.
- A program can do many things, including calculations, sorting names, preparing phone lists, displaying images, validating forms, ad infinitum.
- String literals are rows of characters enclosed in either double or single quotes.
- There are two types of data: primitive and composite.
- An array can hold all your variable values under a single name. And you can access the values by referring to the array name.

6.11 Keywords

Array: An array is a special variable, which can hold more than one value, at a time.

Boolean literals: Boolean literals are logical values that have only one of two values, *true* or *false*.

Data types: Data types specify what kind of data, such as numbers and characters, can be stored and manipulated within a program.

JavaScript: It is a scripting language designed primarily for adding interactivity to Web pages and creating Web applications.

Live Script: Initially java script was called live script.

Null string: An empty set of quotes is called the null string.

SCRIPT tags: In Javascript, SCRIPT tags can be inserted into three places: in between the two BODY tags, in between the two HEAD tags, and as a link to an external file, also in the HEAD section.

String literals: They are rows of characters enclosed in either double or single quotes.

6.12 Review Questions

1. Why scripting languages are used?
2. Write the brief history about JavaScript.
3. What is the difference between Java and JavaScript?
4. Describe about the datatypes in JavaScript.
5. How JavaScript can be tagged with HTML?
6. Describe JavaScript Browsers compatibility.
7. Why JavaScript named as JavaScript?
8. Explain the Typecasting in JavaScript.
9. Describe the arrays in JavaScript.
10. How do you convert numbers between different bases in JavaScript?
11. What is the purpose of backslash in JavaScript?

Answers: Self Assessment

1.	Mocha	2.	LiveScript
3.	JavaScript	4.	web
5.	dynamic	6.	Server
7.	SCRIPT	8.	Elements
9.	page content	10.	<!–
11.	Primitive	12.	String literals
13.	text input	14.	Text area
15.	Type casting	16.	number(value)
17.	array	18.	ID

Notes

6.13 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://en.wikipedia.org/wiki/JavaScript

http://www.homeandlearn.co.uk/JS/javascript_tag_placement.html

Unit 7: Operators in JavaScript

Notes

CONTENTS

Objectives

After studying this unit, you will be able to:

- Discuss about the operators used in JavaScript
- Explain how to use operators in JavaScript?
- Understand what is the precedence of operators in JavaScript?
- Describe what are the expressions in JavaScript?

Introduction

An operator is a symbol (it can also be a word) that performs calculations, comparisons or assignment on one or more values. Much of the work in scripts is done by operators.

7.1 Operators

The most common operators are mathematical operators; +, -, /, * (add, subtract, divide, multiply) for example. Operators can be split into two groups, comparison operators and assignment or 'action' operators. Comparison operators test to see if two variables relate to each other in the specified way.

Example: One variable is a higher number than the other. Other operators perform an action on a variable, such as increasing it by one.

7.1.1 Arithmetic Operators

There are following arithmetic operators supported by JavaScript language:

Assume variable A holds 10 and variable B holds 20 then:

Table 7.1: Arithmetic Operators

Operator	Description	Example
+	Adds two operands	A + B will give 30
-	Subtracts second operand from the first	A - B will give -10
*	Multiply both operands	A * B will give 200
/	Divide numerator by denumerator	B / A will give 2
%	Modulus Operator and remainder of after an integer division	B % A will give 0
++	Increment operator, increases integer value by one	A++ will give 11
--	Decrement operator, decreases integer value by one	A-- will give 9

Notes Addition operator (+) works for Numeric as well as Strings. e.g. "a" + 10 will give "a10"

7.1.2 Bitwise Operators

There are following bitwise operators supported by JavaScript language.

Assume variable A holds 2 and variable B holds 3 then, see Table 7.2.

Table 7.2: Bitwise Operators

Operator	Description	Example
&	Called Bitwise AND operator. It performs a Boolean AND operation on each bit of its integer arguments.	(A & B) is 2 .
|	Called Bitwise OR Operator. It performs a Boolean OR operation on each bit of its integer arguments.	(A | B) is 3.
^	Called Bitwise XOR Operator. It performs a Boolean exclusive OR operation on each bit of its integer arguments. Exclusive OR means that either operand one is true or operand two is true, but not both.	(A ^ B) is 1.

Contd....

Notes

~	Called Bitwise NOT Operator. It is a is a unary operator and operates by reversing all bits in the operand.	(~B) is -4 .
<<	Called Bitwise Shift Left Operator. It moves all bits in its first operand to the left by the number of places specified in the second operand. New bits are filled with zeros. Shifting a value left by one position is equivalent to multiplying by 2, shifting two positions is equivalent to multiplying by 4, etc.	(A << 1) is 4.
>>	Called Bitwise Shift Right with Sign Operator. It moves all bits in its first operand to the right by the number of places specified in the second operand. The bits filled in on the left depend on the sign bit of the original operand, in order to preserve the sign of the result. If the first operand is positive, the result has zeros placed in the high bits; if the first operand is negative, the result has ones placed in the high bits. Shifting a value right one place is equivalent to dividing by 2 (discarding the remainder), shifting right two places is equivalent to integer division by 4, and so on.	(A >> 1) is 1.
>>>	Called Bitwise Shift Right with Zero Operator. This operator is just like the >> operator, except that the bits shifted in on the left are always zero,	(A >>> 1) is 1.

7.1.3 Assignment Operators

The assignment operators supported by JavaScript language are shown in Table 7.3:

Table 7.3: Assignment Operators

Operator	Description	Example
=	Simple assignment operator, Assigns values from right side operands to left side operand	C = A + B will assigne value of A + B into C
+=	Add AND assignment operator, It adds right operand to the left operand and assign the result to left operand	C += A is equivalent to C = C + A
-=	Subtract AND assignment operator, It subtracts right operand from the left operand and assign the result to left operand	C -= A is equivalent to C = C - A
*=	Multiply AND assignment operator, It multiplies right operand with the left operand and assign the result to left operand	C *= A is equivalent to C = C * A

Contd....

/=	Divide AND assignment operator, It divides left operand with the right operand and assign the result to left operand	C /= A is equivalent to C = C / A
%=	Modulus AND assignment operator, It takes modulus using two operands and assign the result to left operand	C %= A is equivalent to C = C % A

Notes Same logic applies to Bitwise operators so they will become like <<=, >>=, >>=, &=, |= and ^=.

7.1.4 Increment Operators

There are also the increment and decrement operators, ++ and —. a++ increments a and returns the old value of a. ++a increments a and returns the new value of a. The decrement operator functions similarly, but reduces the variable instead.

As an example, the last three lines all perform the same task:

```
var a = 1;
a = a + 1;
a += 1;
++a;
```

Pre-and Post-increment Operators

Increment operators may be applied before or after a variable. When they are applied before a variable they are pre-increment operators, and when they are applied after a variable they are post-increment operators.

Caution The choice of which to use changes how they affect operations.

```
// increment occurs before a is assigned to b
var a = 1;
var b = ++a; // a = 2, b = 2;
// increment occurs to c after c is assigned to d
var c = 1;
var d = c++; // c = 2, d = 1;
```

Caution Due to the possibly confusing nature of pre-and post-increment behaviour, code can be easier to read if the increment operators are avoided.

```
// increment occurs before a is assigned to b
var a = 1;
a += 1;
var b = a; // a = 2, b = 2;
// increment occurs to c after c is assigned to d
var c = 1;
var d = c;
c += 1; // c = 2, d = 1;
```

7.1.5 Comparison Operators

There are following comparison operators supported by JavaScript language

Assume variable A holds 10 and variable B holds 20 then, see Table 7.4:

Table 7.4: Comparison Operators

Operator	Description	Example
==	Checks if the value of two operands are equal or not, if yes then condition becomes true.	(A == B) is not true.
!=	Checks if the value of two operands are equal or not, if values are not equal then condition becomes true.	(A != B) is true.
>	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(A > B) is not true.
<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(A < B) is true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(A >= B) is not true.
<=	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	(A <= B) is true.

7.1.6 Logical Operators

Operator	Description
&&	and
||	or
!	not

7.1.7 Other Operators

The Conditional Operator (? :)

There is an operator called conditional operator. This first evaluates an expression for a true or false value and then execute one of the two given statements depending upon the result of the evaluation. The conditional operator has this syntax:

Operator	Description	Example
? :	Conditional Expression	If Condition is true ? Then value X : Otherwise value Y

The *typeof* Operator

The *typeof* is a unary operator that is placed before its single operand, which can be of any type.

Did u know? Its value is a string indicating the data type of the operand.

The *typeof* operator evaluates to “number”, “string”, or “boolean” if its operand is a number, string, or boolean value and returns true or false based on the evaluation.

Here is the list of return values for the *typeof* Operator :

Table 7.5: List of Return Values for the *typeof* Operator

Type	String Returned by typeof
Number	"number"
String	"string"
Boolean	"boolean"
Object	"object"
Function	"function"
Undefined	"undefined"
Null	"object"

Task Give examples of different types of operators.

7.1.8 Operator’s Precedence

The precedence of operators determines the order they are applied when evaluating an expression. You can override operator precedence by using parentheses. The precedence of operators, from lowest to highest is as follows:

```
comma ,
assignment = += -= *= /= %= <<= >>= >>>= &= ^= |=
conditional ?:
logical-or ||
logical-and &&
bitwise-or |
bitwise-xor ^
bitwise-and &
equality == !=
relational < <= > >=
bitwise shift << >> >>>
addition/subtraction + -
multiply/divide * / %
negation/increment ! ~ - ++ —
call, member () [] .
```

Self Assessment

Fill in the blanks:

1. .... operators test to see if two variables relate to each other in the specified way.

2. .... is called Bitwise NOT Operator which is a is a unary operator and operates by reversing all bits in the operand.
3. The operator .... is the multiply AND assignment operator, it multiplies right operand with the left operand and assign the result to left operand.
4. The decrement operator .... the variable.
5. .... operator first evaluates an expression for a true or false value and then execute one of the two given statements depending upon the result of the evaluation.
6. The .... is a unary operator that is placed before its single operand, which can be of any type.
7. The .... of operators determines the order they are applied when evaluating an expression.
8. The operator .... checks if the value of two operands are equal or not, if values are not equal then condition becomes true.
9. Due to the possibly confusing nature of pre-and post-increment behaviour, code can be easier to read if the .... operators are avoided.
10. An operator | is called .... Operator. It performs a Boolean OR operation on each bit of its integer arguments.

7.2 Expressions

An expression is any valid set of literals, variables, operators, and expressions that evaluates to a single value. The value may be a number, a string, or a logical value. Conceptually, there are two types of expressions: those that assign a value to a variable, and those that simply have a value.

Example: The expression

x = 7

It is an expression that assigns x the value 7. This expression itself evaluates to 7. Such expressions use assignment operators. On the other hand, the expression

3 + 4

Simply evaluates to 7; it does not perform an assignment.

Did u know? The operators used in such expressions are referred to simply as operators.

JavaScript has the following kinds of expressions:

- ***Arithmetic:*** evaluates to a number, for example
- ***String:*** evaluates to a character string, for example "Fred" or "234"
- ***Logical:*** evaluates to true or false

The special keyword null denotes a null value. In contrast, variables that have not been assigned a value are undefined, and cannot be used without a run-time error.

Task Make distinction between String expression and logical expression.

7.2.1 Conditional Expression

A conditional expression can have one of two values based on a condition. The syntax is (condition) ? val1 : val2 If condition is true, the expression has the value of val1, otherwise it has the value of val2. You can use a conditional expression anywhere you would use a standard expression.

Example: Status = (age >= 18) ? "adult" : "minor"

This statement assigns the value "adult" to the variable status if age is eighteen or greater. Otherwise, it assigns the value "minor" to status.

Self Assessment

Fill in the blanks:

11. An .... is any valid set of literals, variables, operators, and expressions that evaluates to a single value.
12. The special keyword null denotes a ....
13. A expression can have one of two values based on a ....
14. .... expression evaluates to a character string
15. Variables that have not been assigned a value are undefined, and cannot be used without a .... error.

7.3 Summary

- An operator is a symbol (it can also be a word) that performs calculations, comparisons or assignment on one or more values.
- The most common operators are mathematical operators; +, -, /, *.
- Same logic applies to Bitwise operators so they will become like <<=, >>=, >>=, &=, |= and ^=.
- —. a++ increments a and returns the old value of a. ++a increments a and returns the new value of a.
- Increment operators may be applied before or after a variable.
- There is an operator called conditional operator. This first evaluates an expression for a true or false value and then execute one of the two given statements depending upon the result of the evaluation.
- The *typeof* is a unary operator that is placed before its single operand, which can be of any type.
- The precedence of operators determines the order they are applied when evaluating an expression.

7.4 Keywords

Arithmetic Operators: function as the addition, subtraction, multiplication, division, and modulus operators, and operate very similarly to other languages.

Assignment Operators: assigns a value to a variable.

Bitwise Operators: are kept for parity with the related programming languages but are unlikely to be used in most JavaScript programs.

Conditional Expressions: can have one of two values based on a condition.

Increment Operators: may be applied before or after a variable. When they are applied before a variable they are pre-increment operators, and when they are applied after a variable they are post-increment operators.

Operator: An operator is a symbol (it can also be a word) that performs calculations, comparisons or assignment on one or more values.

Precedence of operators: The precedence of operators determines the order they are applied when evaluating an expression.

Typeof: The *typeof* is a unary operator that is placed before its single operand, which can be of any type.

7.5 Review Questions

1. What is operator? Discuss with example.
2. Differentiate between arithmetic and bitwise operators.
3. Why comparison operators are used?
4. Describe logical operators with the help of example.
5. What is === operator?
6. What does the delete operator do?
7. What is function of Boolean operators in JavaScript?
8. Evaluate "1"+2+4.
9. Solve 2+5+"8".
10. What do you mean by expression in JavaScript?

Answers: Self Assessment

1.	Comparison	2.	~
3.	*=	4.	Reduces
5.	Conditional	6.	*Typeof*
7.	precedence	8.	!=
9.	increment	10.	Bitwise OR
11.	expression	12.	null value

Notes

13. condition
14. String
15. run-time

7.6 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://javascript.about.com/library/blbitop.htm

http://www.howtocreate.co.uk/tutorials/javascript/operators

http://www.quackit.com/javascript/tutorial/javascript_operators.cfm

http://www.tutorialspoint.com/javascript/javascript_operators.htm

https://developer.mozilla.org/en-US/docs/JavaScript/Reference/Operators/Arithmetic_Operators

Unit 8: Programming Constructs in JavaScript

Notes

CONTENTS

Objectives

After studying this unit, you will be able to:

- Discuss about the conditional statements in JavaScript
- Explain looping structures
- Discuss about the functions and its uses

- Discuss about the user-defined function
- Describe the use of dialog boxes in JavaScript

Introduction

JavaScript statements consist of keywords used with the appropriate syntax. A single statement may span multiple lines. Multiple statements may occur on a single line if each statement is separated by a semicolon. These include flow control (if-else, switch), loops (while, do-while, for), and loop control (break, and continue). JavaScript also supports some object-related statements (with, for-in).

Functions are blocks of JavaScript code that perform a specific task and often return a value. A JavaScript function may take zero or more parameters. Parameters are a standard technique via which control data can be passed to a function.

JavaScript provides the ability to pick up user input or display small amount of the text to the user by using dialog boxes. These dialog appear as separate windows and their content depends on the information provided by the user.

8.1 Conditional Statements

Conditional Statements give the JavaScript code you are writing the ability to make decisions or perform single or multiple tasks. These Conditional statements were borrowed from older, more polished languages like C, C++, and Java.

8.1.1 If...Else Statement

The if conditional is used to perform an action "if" the condition is met.

Examine the below syntax example to get a feel for how to build a simple if conditional.

```
if (expression)
statement;
```

The example shows the if keyword followed by an expression within brackets. Within these brackets are the conditions that must be met in order to execute the statement.

If the condition is never met, the statement is never executed. Simply put, the expression must evaluate to true in order for the statement to be executed. Within this simple conditional are an infinite number of variations. Basically, you're letting the script make decisions based on the condition, the expression, you supply.

While performing an action if a value of true is returned from the expression is great, you will eventually have a need for something to happen if the expression evaluates to false. Assigning an action to the false output of the expression involves a very simple addition to the if statement given above. The else keyword is used to supply an action to be taken if the expression is false.

Examine the syntax example below.

```
if (expression) {
  statement if expression is true;
  } else {
  statement If expression is False;
```

Notice the addition not only of the else keyword and an additional statement, but also the use of the opening and closing curly braces - { } . The braces are used to encapsulate the true statement and the false statement, as well as divide the entire statement into its "if" and "else" sections.

Notice also that the basic JavaScript syntax rules haven't been broken – a semi-colon is present after both the true statement and the false statement.

Example:

```
if (80<100)
document.write("The expression has evaluated to true!!");
```

You can see the basic if statement used very simply here. Notice the "less-than" operator used within the conditional expression - "80<100". This if statement simply states that if 80 is less than 100, the statement is considered to be true. Since 80 is actually less than 100, the document.write statement is written to the screen. If the conditional expression were "101<100" (which is read as "if 101 is less than 100"), then nothing would be written to the screen as the conditional expression would have evaluated to false. Use the if statement if you want something to happen only if the conditional expression evaluates to true.

If you'd like an action to be taken when the conditional expression evaluates to false as well as true, use the else addition to provide an action to be taken when the conditional expression returns a false value, as shown below.

```
if (101<100) {
 document.write("The expression has evaluated to true!!");
 } else {
 document.write("The expression has evaluated to false!!");
}
```

The above if/else statement says that if 101 is less than 100, write the first document.write statement to the screen. If 101 isn't less than 100 the second document.write statement is written to the screen. Since 101 is not less than 100, the conditional expression returns false, and the second document.write statement ("The expression has evaluated to false!!") is written to the screen.

So now that you know the basic structure of the if and if/else statements, we'll delve into another aspect – nesting your if/else statements. This nesting structure is used when you have an initial condition to be met with a true or false value, and then you require another set of conditions that you'd like to have tested to acquire your end result.

Examine the below syntax example to understand more clearly what is meant by the term "nesting".

```
if (expression1) {
 statement If expression1 is True;
 } else {
 if (expression2) {
 statement If expression2 is True;
 } else {
 statement If expression2 is False;
 }
}
```

You can see that the second if / else statement is nested within the "false" area of the first if / else statement. If expression1 evaluates to true, then the first document.write statement of the first if / else statement is written to the screen. If expression1 evaluates to false, then the second if / else statement is evaluated. If expression2 evaluates to true, the first document.write statement of the second if / else statement is written to the screen. If expression2 evaluates to false, then the second document.write statement of the second if / else statement is written to the screen. This somewhat complicated (but very useful) method will be used widely within your future JavaScript coding.

Example: To illustrate further, we'll use the nesting of if / else statements in a couple of working examples, shown below.

```
if (99<100) {
 document.write("Expression1 has evaluated to true!!");
 } else {
 if (80<100) {
 document.write("Expression2 has evaluated to true!!");
 } else {
 document.write("Expression2 has evaluated to false!!");
 }
}
```

The example shows the nesting of an if / else statement within another. The first if / else statement shows the less than operator being used to evaluate the weights of two numbers, 99 and 100. Since 99 is indeed less than 100, the conditional expression returns a value of true, and the first document.write statement ("Expression1 has evaluated to true!!") is written to the screen. The false section of the first if / else statement is never consulted, since the conditional expression of the first if / else statement evaluated to true.

Examine the below example, which utilizes the decision making ability of the second, nested, if / else statement.

```
if (101<100) {
 document.write("Expression1 has evaluated to true!!");
 } else {
 if (80<100) {
 document.write("Expression2 has evaluated to true!!");
 } else {
 document.write("Expression2 has evaluated to false!!");
 }
}
```

Since 101 is less than 100, and the conditional expression of the first if / else statement returns false, the second if / else statement is executed. Since 80 is less than 100, the conditional expression returns a value of true and the first document.write statement of the second if / else statement is executed, writing "Expression2 has evaluated to true!!" to the screen.

8.1.2 Switch Statement

The switch case conditional statement is used to test all of the possible outcomes for the application you are designing. Used with the case keyword, the switch statement can give you the control you require, with many actions possible instead of just the two given with an if / else statement. That is, you are not limited to a true or false answer to decide between only two actions. While the true or false decision is still used at a very basic level, you may have as many outcomes as you have a need for.

Examine the syntax example, given below.

```
switch (expression)
 {
 case caseLabel:
 statement;
 break;
 case caseLabel:
```

```
 statement;
 break;
 case caseLabel:
 statement;
 break;
 default:
 statement;
 }
```

Notice the colons following the case and caseLabel keywords. This is absolutely required. Don't leave them out. Don't substitute the colons with semi-colons, or your script won't work. And don't forget the colon following the default keyword near the end of the script.

The switch statement begins with, of course, the switch keyword. Within the brackets following the switch keyword is the expression, which is used to set the parameter that the rest of the script will use as the data to be used to make a decision. Once this decision is made, the script looks for a match among the various case keywords.

Each case keyword has a unique caseLabel associated with it. This is used to define each case statement as a unique and individual entity to your script. Once a case with the matching caseLabel is found, the statement within the case statement is executed.

On the end of the switch statement, you'll see that a "default" keyword was used, with a statement of its own given. This default statement is executed when none of the given cases match the expression given in brackets following the switch keyword.

Notice also that break; statements were used. The break statements used in this way are required to "break" out of the switch/case structure and go on to further actions. It is required so that the rest of the case options below the selected case (as well as the default actions) aren't executed.

Example: To further illustrate the concept, examine the following working example which utilizes the switch/case statement.

```
var varOne = 100;
switch (varOne)
 {
 case 90:
 document.write("The value of varOne is 90");
 break;
 case 100:
 document.write("The value of varOne is 100");
 break;
 case 110:
 document.write("The value of varOne is 110");
 break;
 default:
 document.write("The value of varOne is unknown");
 }
```

The example shows the declaration of a variable, varOne. varOne contains a number, which is 100. This varOne variable is then used as the expression of the switch statement. Now that the script has the expression with to find a match, the several cases given are searched until a case is found that matches the varOne value, which is 100. Since the second label matches the varOne value given as the expression, the statement within that case statement is executed, which is a document.write statement which writes the words "The value of varOne is 100" to the screen.

It should be noted that you aren't limited to using a variable with a static, unchanging value. You may use a variable that changes dynamically with either user driven or script driven actions associated with it to obtain a changing value.

Self Assessment

Fill in the blanks:

1. .... Statements give the JavaScript code you are writing the ability to make decisions or perform single or multiple tasks.
2. The .... conditional statement is used to perform an action "if" the condition is met.
3. The .... conditional statement is used to test all of the possible outcomes for the application you are designing.

8.2 Looping Statements

Often when you write code, you want the same block of code to run over and over again in a row. Instead of adding several almost equal lines in a script we can use loops to perform a task like this.

In JavaScript, there are two different kinds of loops:

- **for** - loops through a block of code a specified number of times
- **while** - loops through a block of code while a specified condition is true

8.2.1 The For Loop

The for loop is used when you know in advance how many times the script should run. Some important points to note are:

- The initialization statements are executed once; only when the *for* loop is encountered.
- After execution of initialization statements, the condition is evaluated.
- After every iteration, the updation statements are executed and then the condition is checked

Syntax

```
for (var=startvalue;var<=endvalue;var=var+increment)
{
code to be executed
}
```

Caution The increment parameter could also be negative, and the <= could be any comparing statement.

Example: Loop example

The example below defines a loop that starts with i=0. The loop will continue to run as long as **i** is less than, or equal to 5. **i** will increase by 1 each time the loop runs.

```
<html>
<body>
<script type="text/javascript">
var i=0;
for (i=0;i<=5;i++)
{
document.write("The number is " + i);
document.write("<br />");
}
</script>
</body>
</html>
```

Output

```
The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
```

Explanation:

This for loop starts with i=0.

As long as **i** is less than, or equal to 5, the loop will continue to run.

i will increase by 1 each time the loop runs.

8.2.2 The While Loop

The while loop loops through a block of code while a specified condition is true.

Syntax

```
while (var<=endvalue)
 {
 code to be executed
 }
```

Did u know? The <= could be any comparing statement.

Example: while loop example

The example below defines a loop that starts with i=0. The loop will continue to run as long as **i** is less than, or equal to 5. **i** will increase by 1 each time the loop runs

```
<html>
<body>
<script type="text/javascript">
var i=0;
while (i<=5)
```

Notes

```
 {
 document.write("The number is " + i);
 document.write("<br />");
 i++;
 }
</script>
</body>
</html>
```

Output

```
The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
```

Explanation:

i is equal to 0.

While **i** is less than, or equal to, 5, the loop will continue to run.

i will increase by 1 each time the loop runs.

8.2.3 The do...while Loop

The do...while loop is a variant of the while loop. This loop will execute the block of code ONCE, and then it will repeat the loop as long as the specified condition is true.

Syntax

```
do
 {
 code to be executed
 }
while (var<=endvalue);
```

Example: do while example

The example below uses a do...while loop. The do...while loop will always be executed at least once, even if the condition is false, because the statements are executed before the condition is tested:

```
<html>
<body>
<script type="text/javascript">
var i=0;
do
 {
document.write("The number is " + i);
 document.write("<br />");
 i++;
 }
while (i<=5);
</script>
</body>
</html>
```

Output

```
The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
```

Explanation:

i equal to 0.

The loop will run

i will increase by 1 each time the loop runs.

While **i** is less than, or equal to, 5, the loop will continue to run.

Task Give examples of while loop and do … while loop.

8.2.4 Break

The break statement will break the loop and continue executing the code that follows after the loop (if any).

Example: Break example

```
<html>
<body>
<script type="text/javascript">
var i=0;
for (i=0;i<=10;i++)
 {
 if (i==3)
 {
 break;
 }
 document.write("The number is " + i);
 document.write("<br />");
 }
</script>
</body>
</html>
```

Notes

Output

The number is 0

The number is 1

The number is 2

Explanation: The loop will break when i=3.

8.2.5 Continue Statements

The continue statement will break the current loop and continue with the next value.

Example: It is showing an example of continue.

```
<html>
<body>
<script type="text/javascript">
var i=0
for (i=0;i<=10;i++)
 {
 if (i==3)
 {
 continue;
 }
 document.write("The number is " + i);
 document.write("<br />");
 }
</script>
</body>
</html>
```

Output

The number is 0

The number is 1

The number is 2

The number is 4

The number is 5

The number is 6

The number is 7

The number is 8

The number is 9

The number is 10

Explanation: The loop will break the current loop and continue with the next value when i=3.

Self Assessment

Fill in the blanks:

4. The .... loop is used when you know in advance how many times the script should run.
5. The .... loop loops through a block of code while a specified condition is true.
6. The .... loop will execute the block of code ONCE, and then it will repeat the loop as long as the specified condition is true.
7. The .... statement will break the loop and continue executing the code that follows after the loop (if any).
8. The .... statement will break the current loop and continue with the next value.

8.3 Functions

To keep the browser from executing a script when the page loads, you can put your script into a function.

A function contains code that will be executed by an event or by a call to the function.

You may call a function from anywhere within a page (or even from other pages if the function is embedded in an external .js file).

Notes Functions can be defined both in the <head> and in the <body> section of a document. However, to assure that a function is read/loaded by the browser before it is called, it could be wise to put functions in the <head> section.

8.3.1 Function Literals

JavaScript 1.2 introduces the concept of *function literals* which is one more new way of defining functions.

A function literal is an expression that defines an unnamed function.

Syntax

The syntax for a *function literal* is much like that of the function statement, except that it is used as an expression rather than as a statement and no function name is required.

```
<script type="text/javascript">
<!–
var variablename = function(Argument List){
 Function Body
 };
//–>
</script>
```

Syntactically, you can specify a function name while creating a literal function as follows:

```
<script type="text/javascript">
<!–
var variablename = function FunctionName(Argument List){
 Function Body
 };
//–>
</script>
```

But this name does not have any significance, so worth not to use it.

8.3.2 Built-in Functions

Javascript, first introduced by Netscape, changed the static fate of the HTML web pages. Before Javascript came into existence, HTML pages were just used to render static content. Inserting Javascript to a web page, can give it a significant degree of interactivity and functionality. It lets you control the behavior of the web browser and how the elements of the web page are displayed. Most web browsers have a built in javascript interpreter. When a browser downloads an html file that contains javascript, the javascript interpreter reacts to any script.

Javascript is an object oriented programming language. It supports the concept of objects in the form of attributes. If an object attribute consists of function, then it is a method of that object, or if an object attribute consists of values, then it is a property of that object.

Example:

var status=document.readyState;

readyState is a property of the document object which can contain values such as "unintialized","loading","interactive","complete" whereas,

document.write("Hello World");

write() is a method of the document object that writes the content "Hello World" on the web page.

There are several Javascript built-in objects such as:

- Number
- String
- RegExp
- Array
- Math
- Date
- Boolean

Each of the above objects hold several built-in functions to perform object related functionality. Apart from these methods, Javascript provides few predefined functions which do not stick to a particular object type but are global.

Example: These global built-in functions are explained below with examples.

isNaN()

isNaN() method determines whether value of a variable is a legal number or not.

```
document.write(isNan(0));
document.write(isNan("Javascript"));
document.write(isNan(-2.45));
document.write(isNan("77"));
document.write(isNan("2012/4/8"));
```

Results:

```
false
true
false
false
true
```

There is a property called NaN of the Number Object which can be used to assign a variable with 'not a number' value.

```
var  year= Number.NaN;
document.write(year);
```

Result:

NaN

isFinite()

As the name indicates, this function is used to find whether a number is a finite legal number.

```
document.write(isFinite("5678"));
document.write(isFinite("ABCD"));
document.write(isFinite("123_456"));
```

Result:

true

false

false

eval()

eval() is used to execute Javascript source code. It evaluates or executes the argument passed to it and generates output.

eval("var number=2;number=number+2;document.write(number)");

Result:

4

Number()

Number() method takes an object as an argument and converts it to the corresponding number value. If the object passed cannot be converted to a number, that is if the object is not in a format to be represented as a number, then it returns NaN(not a number).

var obj1=new String("123");

var obj2=new Boolean("false");

```
var obj3=new Boolean("true");
var obj4=new Date();
var obj5=new String("9191 9999");
document.write(Number(obj1));
document.write(Number(obj2));
document.write(Number(obj3));
document.write(Number(obj4));
document.write(Number(obj5));
```

Result:

```
123
0
1
1342720050291
NaN
```

For date object it returns the number of milliseconds since January 1,1970 UTC.

String()

String() function converts the object argument passed to it to a string value.

```
var obj1=new Boolean(0);
var obj2=new Boolean(1);
var obj3=new Date();
document.write(String(obj1));
document.write(String(obj2));
document.write(String(obj3));
```

Result:

```
false
true
Thu Jul 19 2012 23:28:08 GMT+0530 (India Standard Time)\
```

parseInt()

parseInt() function takes string as a parameter and converts it to integer.

```
document.write(parseInt("50"));
document.write(parseInt("77 days"));
document.write(parseInt("this is 7"));
```

Result:

```
50
77
NaN
```

An optional radix parameter can also be used to specify the number system to be used to parse the string argument. For example,

document.write(parseInt("10",16));

Result:

16

16 is the radix value passed, which means the string should be converted from hexadecimal to decimal value. If the radix is 8, then the string should be converted from octal to decimal value.

If the radix parameter is not specified, then Javascript

- Assumes radix to be 16(hexadecimal), if the string starts with "0x"
- Assumes radix to be 8(octal), if the string starts with 0
- Assumes radix to be 10(decimal), if the string starts with any other number.

parseFloat()

parseFloat() function takes a string as parameter and parses it to a floating point number.

document.write(parseFloat("10.33"));

document.write(parseFloat("15 66 75"));

document.write(parseFloat("this is 77"));

document.write(pareFloat(" 77 "));

Result:

10.33

15

NaN

77

This function allows leading and trailing spaces. If the first character in the string is not a number, then it returns NaN. If the string has more than one set of number separated by delimiters such as spaces, semicolons,commas then it returns only the first set of number before the first delimiter.

escape()

escape() function encodes the string passed to it so that it can be used across any network, say for example in query strings.

document.write(escape("testing escape function!!"));

Result:

testing%20escape%20function%21%21

escape() function leaves digits, latin letters and the characters + - * / . _ @ unchanged and replaces all other characters with ASCII code of the original character preceded by % symbol.

Caution The radix can be any number ranging from 2 to 36 that represents a numeral system.

Did u know? unescape() method is used to decode the encoded strings.

Notes Use of eval method is not advised for the following reasons:

- security risk
- debugging may be difficult
- may result at resource hog since each invocation of eval() creates a new instance of Javascript interpreter.

8.3.3 User-defined Functions

An user-defined function saves us from rewriting the same code again and again and helps us to make our application smaller. The given examples will help you to understand how can you put them to work.

JavaScript has so many built-in functions, besides that you can make your own as per the need. General structure of the user defined function is:

function function_name(par1, par2,..){ statement}

par1, par2 are the name of the parameter, one function could have one, more than one or no parameter. Parameters could be of any datatype.

Example (Simple Function):

```
<html>
<head>
<title>Write your title here </title>
<script type="text/javascript" >
function display()
{
document.write("Hello there");
}
display();
</script>
</head>
</html>
```

Output:

Hello there

8.3.4 Calling Functions

You can call the function from any other JavaScript. After the function is executed, the other script that called it goes on from the call.

```
alert('Example 1: the House');
example(1,'house');
(do more stuff)
```

So this script first generates an alert box, then calls the function and after the function is finished it continues to do the rest of the instructions in the calling code.

8.3.5 Functions Properties

Functions in JavaScript are actually an object. Therefore any created function created using the "function" declaration will have the properties of a function. These properties are:

- arguments - An array of arguments passed to the function. This is an array object which has a property of length which enables the programmer to tell how many arguments (or variables) are passed to the function.
- caller - The name of the function that called the function.
- prototype - Used to make more properties.

8.3.6 Using the Arguments Array

The arguments object is a local variable available within all functions; arguments as a property of Function can no longer be used.

You can refer to a function's arguments within the function by using the arguments object. This object contains an entry for each argument passed to the function, the first entry's index starting at 0.

For example, if a function is passed three arguments, you can refer to the argument as follows:

```
arguments[0]
arguments[1]
arguments[2]
```

The arguments can also be set:

```
arguments[1] = 'new value';
```

The arguments object is not an Array. It is similar to an Array, but does not have any Array properties except length.

Example: It does not have the pop method. However it can be converted to a real Array:

var args = Array.prototype.slice.call(arguments);

If Array generics are available, one can use the following instead:

```
var args = Array.slice(arguments);
```

The arguments object is available only within a function body. Attempting to access the arguments object outside a function declaration results in an error.

You can use the arguments object if you call a function with more arguments than it is formally declared to accept. This technique is useful for functions that can be passed a variable number of arguments. You can use arguments.length to determine the number of arguments passed to the function, and then process each argument by using the arguments object. (To determine the number of arguments declared when a function was defined, use the Function.length property.)

8.3.7 Local Variables

Local variables exist only inside a particular function hence they have ***Local Scope***. *Global variables* on the other hand are present throughout the script and their values can be accessed by any function. Thus, they have ***Global Scope***.

How to initialize a *Local variable*?

Any variable that is initialized inside a function using the *var* keyword will have a local scope. If a variable is initialized inside a function without *var*, it will have a global scope. A local variable can have the same name as a global variable.

```
var a = 10;
disp_a();
function disp_a()
 {
 var a = 20;
 alert("Value of 'a' inside the function " + a);
 }
alert("Value of 'a' outside the function " + a);
```

Self Assessment

Fill in the blanks:

9. To keep the browser from executing a script when the page loads, you can put your script into a ....
10. A .... is an expression that defines an unnamed function.
11. An .... function saves us from rewriting the same code again and again and helps us to make our application smaller.
12. The .... object is a local variable available within all functions.
13. ....variables exist only inside a particular function hence they have Local Scope.

8.4 Dialog Boxes

JavaScript supports three important types of dialog boxes. These dialog boxes can be used to raise and alert, or to get confirmation on any input or to have a kind of input from the users.

Here we will see each dialog box one by one:

8.4.1 Alert Dialog Box

An alert dialog box is mostly used to give a warning message to the users. Like if one input field requires to enter some text but user does not enter that field then as a part of validation you can use alert box to give warning message as follows:

```
<head>
<script type="text/javascript">
<!–
 alert("Warning Message");
//–>
</script>
</head>
```

Nonetheless, an alert box can still be used for friendlier messages. Alert box gives only one button "OK" to select and proceed.

8.4.2 Confirmation Dialog Box

A confirmation dialog box is mostly used to take user's consent on any option. It displays a dialog box with two buttons: **OK** and **Cancel**.

If the user clicks on OK button the window method *confirm()* will return true. If the user clicks on the Cancel button *confirm()* returns false. You can use confirmation dialog box as follows:

```
<head>
<script type="text/javascript">
<!–
  var retVal = confirm("Do you want to continue ?");
  if( retVal == true ){
  alert("User wants to continue!");
        return true;
  }else{
  alert("User does not want to continue!");
        return false;
  }
//–>
</script>
</head>
```

Task Compare and contrast Alert dialog box and Confirmation dialog box.

8.4.3 Prompt Dialog Box

The prompt dialog box is very useful when you want to pop-up a text box to get user input. Thus it enable you to interact with the user. The user needs to fill in the field and then click OK.

This dialog box is displayed using a method called *prompt()* which takes two parameters (i) A label which you want to display in the text box (ii) A default string to display in the text box.

This dialog box with two buttons: **OK** and **Cancel**. If the user clicks on OK button the window method *prompt()* will return entered value from the text box. If the user clicks on the Cancel button the window method *prompt()* returns *null*.

You can use prompt dialog box as follows:

```
<head>
<script type="text/javascript">
<!–
  var retVal = prompt("Enter your name : ", "your name here");
  alert("You have entered : " + retVal );
//–>
</script>
</head>
```

Self Assessment

Fill in the blanks:

14. An .... dialog box is mostly used to give a warning message to the users.
15. The .... dialog box is very useful when you want to pop-up a text box to get user input.

8.5 Summary

- JavaScript statements consist of keywords used with the appropriate syntax.
- Conditional Statements give the JavaScript code you are writing the ability to make decisions or perform single or multiple tasks.
- The switch/case conditional statement is used to test all of the possible outcomes for the application you are designing.
- The for loop is used when you know in advance how many times the script should run.
- The break statement will break the loop and continue executing the code that follows after the loop (if any).
- A function contains code that will be executed by an event or by a call to the function.
- A function literal is an expression that defines an unnamed function.
- An user-defined function saves us from rewriting the same code again and again and helps us to make our application smaller.
- Dialog boxes can be used to raise and alert, or to get confirmation on any input or to have a kind of input from the users.

8.6 Keywords

Alert dialog box: An alert dialog box is mostly used to give a warning message to the users.

Break statement: It will break the loop and continue executing the code that follows after the loop (if any).

Conditional Statements: It gives the JavaScript code you are writing the ability to make decisions or perform single or multiple tasks.

Confirmation dialog box: A confirmation dialog box is mostly used to take user's consent on any option.

Function literal: A function literal is an expression that defines an unnamed function.

Switch case: Conditional statement is used to test all of the possible outcomes for the application you are designing.

User-defined function: An user-defined function saves us from rewriting the same code again and again and helps us to make our application smaller.

While loop: This loop loops through a block of code while a specified condition is true.

8.7 Review Questions

1. Differentiate between functions and statements in JavaScript.
2. What is the use of if-else and switch statements?
3. Discuss break-continue statements with example.
4. What are the built-in functions in JavaScript?
5. Differentiate between built-in functions and user-defined functions.
6. What is the difference between for and while loop statements?

7. What is the use of dialog boxes in JavaScript?
8. Differentiate between alert and prompt dialog boxes in JavaScript.
9. What is the use of confirm dialog box in JavaScript?
10. What are the differences between confirm and alert dialog boxes in JavaScript?

Answers: Self Assessment

1.	Conditional	2.	If
3.	switch case	4.	For
5.	while	6.	do...while
7.	break	8.	Continue
9.	function	10.	function literal
11.	user-defined	12.	Arguments
13.	Local	14.	Alert
15.	prompt		

8.8 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://www.quirksmode.org/js/function.html

http://www.roseindia.net/tutorial/javascript/javascriptbasics/JavaScript-User-Defined-Functions.html

http://www.tutorialspoint.com/javascript/javascript_function_literals.htm

http://www.webdevelopersnotes.com/tutorials/javascript/global_local_variables_scope_javascript.php3

https://developer.mozilla.org/en-US/docs/JavaScript/Reference/Functions_and_function_scope/arguments

http://www.programming-free.com/2012/07/javascript-built-in-functions-with.html#.UYFBhVIg7IU

Notes

Unit 9: DOM Model

CONTENTS

Objectives

After studying this unit, you will be able to:

- Understand DOM Model
- Discuss Objects in HTML

Introduction

The Document Object Model (DOM) is a programming API for HTML and XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated. In the DOM specification, the term "document" is used in the broad sense – increasingly, XML is being used as a way of representing many different kinds of information that may be stored in diverse systems, and much of this would traditionally be seen as data rather than as documents. Nevertheless, XML presents this data as documents, and the DOM may be used to manage this data.

With the Document Object Model, programmers can create and build documents, navigate their structure, and add, modify, or delete elements and content. Anything found in an HTML or XML document can be accessed, changed, deleted, or added using the Document Object Model, with a few exceptions – in particular, the DOM interfaces for the internal subset and external subset have not yet been specified.

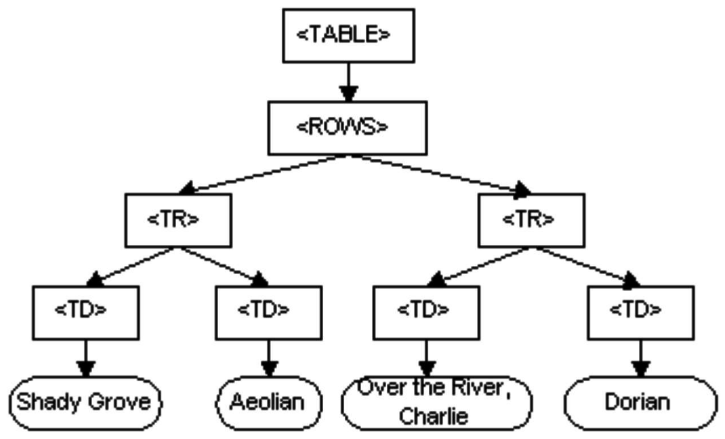
9.1 DOM Model – An Overview

The Document Object Model is a programming API for documents. The object model itself closely resembles the structure of the documents it models. For instance, consider this table, taken from an HTML document:

```
<TABLE>
<ROWS>
<TR>
<TD>Shady Grove</TD>
<TD>Aeolian</TD>
</TR>
<TR>
<TD>Over the River, Charlie</TD>
<TD>Dorian</TD>
</TR>
</ROWS>
</TABLE>
```

The Document Object Model represents this table like Figure 9.1.

Figure 9.1: DOM Representation of the Example Table



The name "Document Object Model" was chosen because it is an "object model" is used in the traditional object oriented design sense: documents are modeled using objects, and the model encompasses not only the structure of a document, but also the behavior of a document and the objects of which it is composed. In other words, the nodes in the above diagram do not represent a data structure, they represent objects, which have functions and identity. As an object model, the Document Object Model identifies:

- the interfaces and objects used to represent and manipulate a document
- the semantics of these interfaces and objects – including both behavior and attributes
- the relationships and collaborations among these interfaces and objects

The Document Object Model currently consists of two parts, DOM Core and DOM HTML. The DOM Core represents the functionality used for XML documents, and also serves as the basis for DOM HTML.

Caution All DOM implementations must support the interfaces listed as "fundamental" in the Core specification; in addition, XML implementations must support the interfaces listed as "extended" in the Core specification.

Did u know? The Level 1 DOM HTML specification defines additional functionality needed for HTML documents.

9.1.1 Objects of Desire

The objects are self-contained bundles of data. Variables associated with an object are called *properties* of the object. Functions that can be executed by an object are called *methods* of the object.

There are three kinds of objects in JavaScript:

- User-defined objects created from scratch by the programmer.
- Native objects like Array, Math, and Date, which are built in to JavaScript.
- Host objects that are provided by the browser.

From the earliest days of JavaScript, some very important host objects have been made available for scripting. The most fundamental of these is the window object.

The window object is nothing less than a representation of the browser window itself. The properties and methods of the window object are often referred to as the Browser Object Model (although perhaps Window Object Model would be more semantically correct).

The Browser Object Model has methods like window.open and window.blur. These methods, incidentally,are responsible for all those annoying pop-up and pop-under windows that have plagued the Web.

Fortunately, we won't be dealing with the Browser Object Model very much. Instead, we'll focus on what's inside the browser window. The object that handles the contents of a web page is the document object. For the rest of this book, we're going to be dealing almost exclusively with the properties and methods of the document object.

9.1.2 Dial M for Model

The *M* in DOM stands for Model, but it could just as easily stand for Map. A model, like a map, is a representation of something. A model train represents a real train. A street map of a city represents the real city. The DOM represents the web page that's currently loaded in the browser window. The browser provides a map (or model) of the page. You can use JavaScript to read this map.

Maps make use of conventions like direction, contours, and scale. In order to read a map, you need to understand these conventions – and it's the same with the DOM. The most important convention used by the DOM is the representation of a document as a tree. More specifically, the document is represented as a family tree.

In order to gain information from the model, you need to understand what conventions are being used to represent the document.

Task Analyse the uses of Dial M for Model.

9.1.3 Nodes

A Node is an interface from which a number of DOM types inherit, and allows these various types to be treated (or tested) similarly, the Node object represents a single node in the document tree.

A node can be an element node, an attribute node, a text node, or any other of the node types .Notice that while all objects inherits the Node properties/methods for dealing with parents and children, not all objects can have parents or children.

Example: Text nodes may not have children, and adding children to such nodes results in a DOM error.

Self Assessment

Fill in the blanks:

1. The .... is a programming API for documents.
2. The DOM Core represents the functionality used for .... documents.
3. Variables associated with an object are called .... of the object.
4. The *M* in DOM stands for Model, but it could just as easily stand for ....
5. A .... is an interface from which a number of DOM types inherit, and allows these various types to be treated (or tested) similarly.

9.2 Objects in HTML

The Object object represents an HTML object element.

The <object> tag is used to include objects such as images, audio, videos, Java applets, ActiveX, PDF, and Flash into a webpage.

9.2.1 Document Object

The Document object is the root of a document tree. The Document object gives us access to the document's data. Since element nodes, text nodes, attributes, comments, etc. cannot exist outside the document, the Document object contains methods to create these objects. All Node objects have a ownerDocument property which associates them with the Document where they were created.

Object is supported in all major browsers.

Document Object Properties

The "DOM" column indicates in which DOM Level the property was introduced.

Property	Description	DOM
doctype	Returns the Document Type Declaration associated with the document	1
documentElement	Returns the Document Element of the document (the HTML element)	1

Contd....

documentURI	Sets or returns the location of the document	3
domConfig	Returns the configuration used when normalize Document() is invoked	3
implementation	Returns the DOMImplementation object that handles this document	1
inputEncoding	Returns the encoding, character set, used for the document	3
strictErrorChecking	Sets or returns whether error-checking is enforced or not	3
xmlEncoding	Returns the XML encoding of the XML document	3
xmlStandalone	Sets or returns whether the XML document is standalone or not	3
xmlVersion	Sets or returns the XML version of the XML document	3

Document Object Methods

The “DOM” column indicates in which DOM Level the method was introduced.

Method	Description	DOM
adoptNode(node)	Adopts a node from another document to this document. Returns the adopted node	3
createAttribute()	Creates an attribute node	1
createAttributeNS(*URI*,*name*)	Creates an attribute with the specified name and namspaceURI	2
createCDATASection()	Creates a CDATA node with the specified text. For XML DOM only	1
createComment()	Creates a Comment node with the specified text	1
createDocumentFragment()	Creates an empty DocumentFragment node	1
createElement()	Creates an Element node	1
createElementNS()	Creates an element with the specified namespace	2
createEntityReference()	Creates an EntityReference node. For XML DOM only	1
createProcessingInstruction()	Creates an EntityReference node. For XML DOM only	1
createTextNode()	Creates a Text node	1
getElementById()	Returns the element that has the ID attribute with the specified value	2
getElementsByTagName()	Returns a NodeList containing all elements with the specified tagname	1
getElementsByTagNameNS()	Returns a NodeList containing all elements with the specified namespaceURI and tagname	2
importNode()	Imports a node from another document	2
normalizeDocument()	Removes empty Text nodes, and joins adjacent nodes	3
renameNode()	Renames the specified node	

9.2.2 Event Object

By using JavaScript, we have the ability to create dynamic web pages. Events are actions that can be detected by JavaScript.

Every element on a web page has certain events which can trigger a JavaScript.

Example: We can use the onClick event of a button element to indicate that a function will run when a user clicks on the button.

We define the events in the HTML tags.

Examples of events:

- A mouse click
- A web page or an image loading
- Mousing over a hot spot on the web page
- Selecting an input field in an HTML form
- Submitting an HTML form
- A keystroke

Notes Events are normally used in combination with functions, and the function will not be executed before the event occurs!

- ***onLoad and onUnload:*** The onLoad and onUnload events are triggered when the user enters or leaves the page. The onLoad event is often used to check the visitor's browser type and browser version, and load the proper version of the web page based on the information.

Did u know? Both the onLoad and onUnload events are also often used to deal with cookies that should be set when a user enters or leaves a page.

Example: You could have a popup asking for the user's name upon his first arrival to your page. The name is then stored in a cookie. Next time the visitor arrives at your page, you could have another popup saying something like: "Welcome upendra!".

- ***onFocus, onBlur and onChange:*** The onFocus, onBlur and onChange events are often used in combination with validation of form fields.

Example: Below is an example of how to use the onChange event. The checkEmail() function will be called whenever the user changes the content of the field:

```
<input type="text" size="30" id="email" onchange="checkEmail()">
```

- ***onSubmit:*** The onSubmit event is used to validate ALL form fields before submitting it.

Example: Below is an example of how to use the onSubmit event. The checkForm() function will be called when the user clicks the submit button in the form.

```
<form method="post" action="xxx.htm" onsubmit="return checkForm()">
```

Notes

Caution If the field values are not accepted, the submit should be cancelled.

The function checkForm() returns either true or false. If it returns true the form will be submitted, otherwise the submit will be cancelled:

```
onMouseOver and onMouseOut
onMouseOver and onMouseOut are often used to create "animated" buttons.
```

Example: Below is an example of an onMouseOver event. An alert box appears when an onMouseOver event is detected:

<a href="a.html" onmouseover="alert('An onMouseOver event');return false</a>

Task Compare and contrast onLoad and onUnload events.

Event Association

Events are associated with HTML tags. The definitions of the events described below are as follows:

- ***abort*** - A user action caused an abort of an image or document load.
- ***blur*** - A frame set, document, or form object such as a text field loses the focus for input.
- ***click*** - Happens when a link or image map is clicked on.
- ***change*** - Happens when a form field is changed by the user and it loses the focus.
- ***error*** - An error happened loading a image or document.
- ***focus*** - A frame set, document, or form object such as a text field gets the focus for input.
- ***load*** - The event happens when an image or HTML page has completed the load process in the browser.
- ***mouseOut*** - The event happens when the mouse is moved from on top of a link or image map
- ***mouseOver*** - The event happens when the mouse is placed on a link or image map.
- ***reset*** - The user reset the object which is usually a form.
- ***submit*** - The user submitted an object which is usually a form.
- ***unload*** - The object such as a framesed or HTML document was exited by the user.

The events for each HTML tag are as follows:

- <A>
  - click (onClick)
  - mouseOver (onMouseOver)
  - mouseOut (onMouseOut)

- <AREA>
  - mouseOver (onMouseOver)
  - mouseOut (onMouseOut)
- <BODY>
  - blur (onBlur)
  - error (onError)
  - focus (onFocus)
  - load (onLoad)
  - unload (onUnload)
- <FORM>
  - submit (onSubmit)
  - reset (onReset)
- <FRAME>
  - blur (onBlur)
  - focus (onFocus)
- <FRAMESET>
  - blur (onBlur)
  - error (onError)
  - focus (onFocus)
  - load (onLoad)
  - unload (onUnload)
- <IMG>
  - abort (onAbort)
  - error (onError)
  - load (onLoad)
- <INPUT TYPE = "button">
  - click (onClick)
- <INPUT TYPE = "checkbox">
  - click (onClick)
- <INPUT TYPE = "reset">
  - click (onClick)
- <INPUT TYPE = "submit">
  - click (onClick)
- <INPUT TYPE = "text">
  - blur (onBlur)

Notes

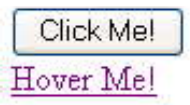
- focus (onFocus)
- change (onChange)
- select (onSelect)

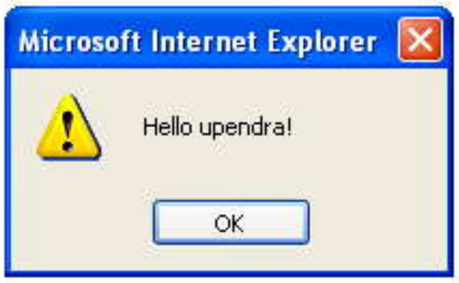
- <SELECT>
  - blur (onBlur)
  - focus (onFocus)
  - change (onChange)
- <TEXTAREA>
  - blur (onBlur)
  - focus (onFocus)
  - change (onChange)
  - select (onSelect)

Example:

```
<html>
<head>
<script type="text/javascript">
function popup()
       {
       alert("Hello upendra!")
       }
</script>
</head>
<body>
<input type="button" value="Click Me!" onclick="popup()"><br />
<a href="#" onmouseover="" onMouseout="popup()">
Hover Me!</a>
</body>
</html>
```

Output





Hello upendra!

Notes

9.2.3 Element Object

The following properties, and methods can be used on all HTML elements.

Notes All elements can also use the properties/methods of the Node object and the Element object.

Properties

Property	Description	W3C
accessKey	Sets or returns an accesskey for an element	Yes
className	Sets or returns the class attribute of an element	Yes
clientHeight	Returns the viewable height of the content on a page (not including borders, margins, or scrollbars)	No
clientWidth	Returns the viewable width of the content on a page (not including borders, margins, or scrollbars)	No
dir	Sets or returns the text direction of an element	Yes
id	Sets or returns the id of an element	Yes
innerHTML	Sets or returns the HTML contents (+text) of an element	Yes
lang	Sets or returns the language code for an element	Yes
offsetHeight	Returns the height of an element, including borders and padding if any, but not margins	No
offsetLeft	Returns the horizontal offset position of the current element relative to its offset container	No
offsetParent	Returns the offset container of an element	No
offsetTop	Returns the vertical offset position of the current element relative to its offset container	No
offsetWidth	Returns the width of an element, including borders and padding if any, but not margins	No
scrollHeight	Returns the entire height of an element (including areas hidden with scrollbars)	No
scrollLeft	Returns the distance between the actual left edge of an element and its left edge currently in view	No
scrollTop	Returns the distance between the actual top edge of an element and its top edge currently in view	No
scrollWidth	Returns the entire width of an element (including areas hidden with scrollbars)	No
style	Sets or returns the style attribute of an element	Yes
tabIndex	Sets or returns the tab order of an element	Yes
title	Sets or returns the title attribute of an element	Yes

Notes

HTML Element Methods

Method	Description	W3C
toString()	Converts an element to a string	Yes

Self Assessment

Fill in the blanks:

6. The .... tag is used to include objects such as images, audio, videos, Java applets, ActiveX, PDF, and Flash into a webpage.
7. The Document object is the root of a ....
8. .... is used to create an attribute node.
9. .... method is used to import a node from another document.
10. .... are actions that can be detected by JavaScript.
11. Every element on a .... has certain events which can trigger a JavaScript.
12. The .... event is often used to check the visitor's browser type and browser version, and load the proper version of the web page based on the information.
13. The .... event is used to validate ALL form fields before submitting it.
14. onMouseOver and onMouseOut are often used to create "..........................." buttons.
15. .... is a frame set, document, or form object such as a text field loses the focus for input.

9.3 Summary

- The Document Object Model (DOM) is a programming API for HTML and XML documents.
- The Document Object Model currently consists of two parts, DOM Core and DOM HTML. The DOM Core represents the functionality used for XML documents, and also serves as the basis for DOM HTML.
- The objects are self-contained bundles of data.
- Object tag is used to include objects such as images, audio, videos, Java applets, ActiveX, PDF, and Flash into a webpage.
- The M in DOM stands for Model, but it could just as easily stand for Map.
- A Node is an interface from which a number of DOM types inherit, and allows these various types to be treated (or tested) similarly.
- The Document object is the root of a document tree. The Document object gives us access to the document's data.
- By using JavaScript, we have the ability to create dynamic web pages. Events are actions that can be detected by JavaScript.

9.4 Keywords

abort : It is a user action caused an abort of an image or document load.

blur : It is a frame set, document, or form object such as a text field loses the focus for input.

change: It happens when a form field is changed by the user and it loses the focus.

click: It happens when a link or image map is clicked on.

DOM Model: The Document Object Model is a programming API for documents.

error: An error happened loading a image or document.

Node: It is an interface from which a number of DOM types inherit, and allows these various types to be treated (or tested) similarly.

Object: This tag is used to include objects such as images, audio, videos, Java applets, ActiveX, PDF, and Flash into a webpage.

9.5 Review Questions

1. What is DOM?
2. Explain the DOM interface.
3. How do you define DOM?
4. How do you define nodes in HTML DOM?
5. How do you define node tree in HTML DOM?
6. What are the properties and methods of HTML DOM?
7. Write a short note on HTML Element Properties.
8. Describe the properties of Document Object.
9. Illustrate the representation of DOM with example.
10. Make distinction between onMouseOver and onMouseOut.

Answers: Self Assessment

1.	Document Object Model	2.	XML
3.	properties	4.	Map
5.	Node	6.	<object>
7.	document tree	8.	createAttribute()
9.	importNode()	10.	Events
11.	web page	12.	onLoad
13.	onSubmit	14.	Animated
15.	blur		

Notes

9.6 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://books.google.co.in/books?id=Yg9BXNlAwloC&pg=PA32&lpg=PA32&dq=dial+m+in+dom+model&source=bl&ots= X3Ac6Ftw5x&sig= 4TIsRZz4-SgE6DT1WXPNRieZDd0&hl=en&sa=X&ei=alKBU bidFcysrAfA2YCYAw&ved=0CDA Q6AEwAg#v=onepage &q=dial%20m%20in%20dom%20model&f=false

http://www.w3schools.com/dom/dom_node.asp

http://www.w3schools.com/jsref/dom_obj_core_document.asp

Unit 10: Browser Objects

CONTENTS

Objectives

After studying this unit, you will be able to:

- Discuss the Window objects
- Explain the Navigator objects
- Describe the History objects
- Discuss the Location objects
- Explain the Document object

Introduction

All browsers are split into different parts (objects) that can be accessed using Javascript. Collectively, these parts are known as the Browser Object Model, or the BOM. At the very top of this browser hierarchy is the Window object. This represents the entire browser, with its toolbars, menus, status bar, the page itself, and a whole lot more besides. Effectively, the Window is the browser.

10.1 Window Objects

The window object, which is a top-level object in Client Side JavaScript, represents a window or a frame (within a frameset). Since window object is a top-level object, it contains other objects like 'document', 'history' etc. within it.

Example: Window.document.frames[i], where i is 1,2,3....., refers to the index of the frame residing in the current window.

For a top-level window, the parent and top properties refer to the window itself. For frame, the top refers to the topmost browser window, and parent refers to the parent window of the current window.

If a frame has name and src attribute you can refer to that frame from a sibling frame by using parent.frameName or parent.frames[i], where i is 1,2,3.......

Did u know? A window object is opened with window.open() and closed with window.close(), if the window does not refer to a frame.

In the subsequent pages we have discussed window object in detail with lots of examples.

10.1.1 JavaScript Window Objects Property

Name	Description	Version
closed	Determine whether a window is closed or not.	Implemented in JavaScript 1.1
defaultStatus	Retrieves the default message displayed in the window's status bar at the bottom.	Implemented in JavaScript 1.0
document	Contains information about the current document.	Implemented in JavaScript 1.0
innerHeight	Specifies the vertical dimension(height of the window), in pixels.	Implemented in JavaScript 1.2
innerWidth	Specifies the horizontal dimension(width of the document)), in pixels.	Implemented in JavaScript 1.2
location	Specify the information of the current URL.	Implemented in JavaScript 1.0
locationbar	Test whether the location bar is visible is or not.	Implemented in JavaScript 1.2
menubar	Refers the browser window's menu bar.	Implemented in JavaScript 1.2
name	Get the unique name of a specified window.	Implemented in JavaScript 1.0
opener	Specifies the window of the calling document.	Implemented in JavaScript 1.1

Contd....

outerHeight	Specifies the vertical dimension, in pixels, of the window's outside boundary.	Implemented in JavaScript 1.2
outerWidth	Specifies the horizontal dimension of the window's outside boundary.	Implemented in JavaScript 1.2
pageXOffset	Refers the current x-position of a page in a window.	Implemented in JavaScript 1.2
pageYOffset	Refers the current y-position of a page in a window.	Implemented in JavaScript 1.2
personalbar	Refers the browser window's personal bar.	Implemented in JavaScript 1.2
screenX	Specify the x coordinate of the left edge of the window.	Implemented in JavaScript 1.2
screenY	Refers the y-coordinate of the upper edge of a window.	Implemented in JavaScript 1.2
scrollbars	Refers the scroll bars of the browser window..	Implemented in JavaScript 1.2
status	Specifies a string displayed in the browser status bar at the bottom of the window.	Implemented in JavaScript 1.0
statusbar	Refers the status bar of the browser window.	Implemented in JavaScript 1.2
toolbar	Refers the toolbar of the browser window.	Implemented in JavaScript 1.2

Task Compare and contrast innerHeight and innerWidth property.

10.1.2 JavaScript Window Objects Methods

Name	Description	Version
alert	Displays an Alert dialog box with a message and an OK button.	Implemented in JavaScript 1.0
atob	Decodes a string of encoding data using base-64 encoding.	Implemented in JavaScript 1.2
back	Takes the browser to the previous URL in the current history list.	Implemented in JavaScript 1.2
blur	Removes focus from a specific window or frame.	Implemented in JavaScript 1.0
btoa	Convert a given string to a encoded data (using base-64 encoding) string.	Implemented in JavaScript 1.2
clearInterval	Cancels a timeout that which was set with the setInterval method.	Implemented in JavaScript 1.2
clearTimeout	Cancels a timeout which was set with the setTimeout method.	Implemented in JavaScript 1.0

Contd....

Notes

close	Closes the specified window.	Implemented in JavaScript 1.0
confirm	Displays a Confirm dialog box with the specified message and two buttons OK and Cancel.	Implemented in JavaScript 1.0
find	Search the specified text string in the contents of the specified window.	Implemented in JavaScript 1.2
focus	Gives focus to a specific window.	Implemented in JavaScript 1.1
forward	Takes the browser to the next URL in the current history list.	Implemented in JavaScript 1.2
home	Returns the browser to the user's specified home page.	Implemented in JavaScript 1.2
moveBy	Move the window relative to it's current position in pixels.	Implemented in JavaScript 1.2
moveTo	Move the top-left corner of the window to the given screen coordinates.	Implemented in JavaScript 1.2
open	Opens a new web browser window.	Implemented in JavaScript 1.0
print	Prints the contents of the window.	Implemented in JavaScript 1.2
prompt	Displays a Prompt dialog box including a message and an input field.	Implemented in JavaScript 1.0
resizeBy	Resizes an entire window by moving the window's bottom-right corner by the specified number of pixels.	Implemented in JavaScript 1.2
resizeTo	Resizes an entire window to the specified pixel dimensions.	Implemented in JavaScript 1.2
scrollBy	Scrolls the specified window by the number of pixels passed to the method.	Implemented in JavaScript 1.2
scrollTo	Scrolls the viewing area of the window so that the specified point becomes the top-left corner.	Implemented in JavaScript 1.2
setInterval	Either evaluates an expression or calls a function after every specified number of	Implemented in JavaScript 1.2

Self Assessment

Fill in the blanks:

1. For a .... window, the parent and top properties refer to the window itself.
2. .... method decodes a string of encoding data using base-64 encoding.
3. ....evaluates an expression or calls a function after a specified number of milliseconds.

10.2 Navigator Objects

Contains information about the version, mimetype and what plug-ins users have installed of Navigator in use.

Notes

Caution It cannot be created by user. It is automatically created by the javascript runtime engine.

10.2.1 Navigator Objects – Properties

Name	Description	Version
appCodeName	Specifies the code name of the browser.	Implemented in JavaScript 1.0
appName	Specifies the name of the browser.	Implemented in JavaScript 1.0
appVersion	Specifies version information for the Navigator.	Implemented in JavaScript 1.0
language	Indicates the translation of the Navigator being used.	Implemented in JavaScript 1.2
mimeTypes	An array of all MIME types supported by the client.	Implemented in JavaScript 1.1
platform	Indicates the machine type for which the Navigator was compiled.	Implemented in JavaScript 1.2
plugins	An array of all plug-ins currently installed on the client.	Implemented in JavaScript 1.1
userAgent	Specifies the user-agent header.	Implemented in JavaScript 1.0

10.2.2 Navigator Objects – Methods

Name	Description	Version
javaEnabled	Tests whether Java is enabled.	Implemented in JavaScript 1.1
taintEnabled	Specifies whether data tainting is enabled.	Implemented in JavaScript 1.1

Self Assessment

Fill in the blanks:

4. .... contains information about the version, mimetype and what plug-ins users have installed of Navigator in use.

5. .... property specifies version information for the Navigator.

6. .... property tests whether Java is enabled.

10.3 History Objects

The JavaScript History Object is property of the window object.

10.3.1 History Objects – Properties

The JavaScript history object contains the following properties:

- *current* - The current document URL.
- *length* - The number of entries in the history object.

Notes

- *next* - The URL of the next document in the history object.
- *previous* - The URL of the last document in the history object.

10.3.2 History Objects – Methods

The Methods of JavaScript history objects are as follows:

- *back()* - Go to the previous URL entry in the history list. This does the same thing as the browser back button.
- *forward()* - Go to the next URL entry in the history list. This does the same thing as the browser forward button. This is only effective when there is a next document in the history list. The back function or browser back button must have previously been used for this function to work.
- *go(relPos | string)* - This function will accept an integer or a string. If an integer is used, the browser will go forward or back (if the value is negative) the number of specified pages in the history object (if the requested entry exists in the history object).

Self Assessment

Fill in the blanks:

7. The JavaScript .... is property of the window object.
8. .... function will accept an integer or a string.
9. The .... property specifies the number of entries in the history object.

10.4 Location Objects

The JavaScript location object is a property of the window object. It can be used to control the web page displayed by the browser.

10.4.1 Location Objects – Properties

JavaScript location object has the follwing properties:

- *hash* - The URL anchor part including the leading hash mark if one exists This is the part of the URL that is used to point to point to a particular part of a page where a named anchor is.

Notes The hash is the part containing the # sign that points to the particular page location.

- *host* - The URL hostname and port. The URL http://abc.com/index.html has the host value of ctdp.tripod.com:80. The colon and port is only included when specified. The URL http://ctdp.abc.com/index.html has the host value of ctdp.tripod.com.
- *hostname* - The URL hostname section
- *href* - The entire URL. The following code will load the home CTDP page: location.href = "http://ctdp.tripod.com/" The following code will display the URL of the current page: document.write(location.href)
- *pathname* - The URL pathname section
- *port* - The URL port section.
- *protocol* - The URL protocol section including the colon after the protocol name. The values are normally http: or file:.

The following JavaScript code may be used to identify the source of the URL.

```
switch (window.location.protocol)
{
case "http:":
document.write("From Web<BR>\n")
break
case "file:":
document.write("From Local computer<BR>\n")
break
default:
document.write("Unknown Source<BR>\n")
break
}
```

- ***search -*** The URL query string section. This is the section after and including the question mark.
- ***target -*** The URL link's target name.

Task Analyse the use of Location objects.

10.4.2 Location Objects – Methods

The following method are used in location objects:

- ***reload()*** - The current window document is reloaded. If a value of true is passed to the reload function, the reload is forced to do an HTTP GET of the document.

Notes This is not normally done but is useful when you think the server contents may be different from your cache.

- ***replace(URL)*** - Requires a URL as a parameter. It loads the document at the URL on top of the current document.

Did u know? A new entry is not placed in the history object.

Self Assessment

Fill in the blanks:

10. The JavaScript .... is a property of the window object.
11. .... is the part of the URL that is used to point to point to a particular part of a page where a named anchor is.
12. .... property specifies the URL hostname and port.

Notes

10.5 Document Object

The JavaScript Document object is the container for all HTML HEAD and BODY objects associated within the HTML tags of an HTML document.

10.5.1 Document Object – Properties

Document object has the following properties:

- ***alinkColor*** - The color of active links.
- ***bgColor*** - Sets the background color of the web page. It is set in the <body> tag. The following code sets the background color to white.

  document.bgColor = "#FFFFFF"
- ***cookie*** - Used to identify the value of a cookie.
- defaultCharset
- ***domain*** - The domain name of the document server.
- ***embeds*** - An array containing all the plugins in a document.
- ***fgColor*** - The text color attribute set in the <body> tag.
- ***FileCreatedDate*** - Use this value to show when the loaded HTML file was created.
- ***fileModifiedDate*** - Use this value to show the last change date of the loaded HTML file
- fileSize
- fileUpdatedDate
- ***lastModified*** - The date the file was modified last.
- ***layers*** - An array containing all the layers in a document.
- ***linkColor*** - The color of HTML links in the document. It is specified in the <body> tag.
- location
- mimeType
- nameProp
- protocol
- readyState
- ***referrer*** - The Universal Resource Locator (URL) of the document that we got the link to the present document from.
- security
- ***title*** - The name of the current document as described between the header TITLE tags.
- ***URL*** - The location of the current document.
- ***vlinkColor*** - The color of visited links as specified in the <body> tag/

10.5.2 Document Object – Methods

The following methods are used in document object:

- ***clear() -*** This is depreciated.
- ***close() -*** Closes an output stream that was used to create a document object.
- ***contextual() -*** It can be used to specify stype of specific tags.

The following example specified that text in blockquotes is to be blue:

document.contextual(document.tags.blockquote).color = "blue";

Multiple styles may be specified in the contextual method to set the value of text in a H3 tag that is underlined to the color blue, for example.

document.contextual(document.tags.H3, document.tags.U).color = "blue";

- elementFromPoint(x, y) - Returns the object at point x, y in the HTML document.
- getSelection() - Get the selected text (if any is selected).
- open([mimeType]) - Opens a new document object with the optional MIME type.
- write(expr1[,expr2...exprN]) - Add data to a document. Writes the values passed to the write function to the document.

  document.write("<H3>") document.writeln("This is a Header") document.write("</H3>")
- writeln(expr1[,expr2...exprN]) - Adds the passed values to the document appended with a new line character.

Self Assessment

Fill in the blanks:

13. The JavaScript .... is the container for all HTML HEAD and BODY objects associated within the HTML tags of an HTML document.
14. .... property sets the background color of the web page.
15. .... property specifies the text color attribute set in the <body> tag.

10.6 Summary

- All browsers are split into different parts (objects) that can be accessed using Javascript.
- The window object, which is a top-level object in Client Side JavaScript, represents a window or a frame (within a frameset).
- For a top-level window, the parent and top properties refer to the window itself.
- A window object is opened with window.open() and closed with window.close(), if the window does not refer to a frame.
- The JavaScript History Object is property of the window object.
- The JavaScript Document object is the container for all HTML HEAD and BODY objects associated within the HTML tags of an HTML document.
- Navigator Objects contains information about the version, mimetype and what plug-ins users have installed of Navigator in use.
- The JavaScript location object is a property of the window object. It can be used to control the web page displayed by the browser.

10.7 Keywords

contextual(): It can be used to specify stype of specific tags.

Cookie: This property is used to identify the value of a cookie.

Document Object: The JavaScript Document object is the container for all HTML HEAD and BODY objects associated within the HTML tags of an HTML document.

go(relPos | string): This function will accept an integer or a string.

History Object: The JavaScript History Object is property of the window object.

Location Object: The JavaScript location object is a property of the window object. It can be used to control the web page displayed by the browser.

Navigator Objects: It Contains information about the version, mimetype and what plug-ins users have installed of Navigator in use.

Window object: The window object, which is a top-level object in Client Side JavaScript, represents a window or a frame (within a frameset).

10.8 Review Questions

1. What is History object function in JavaScript?
2. Write a short note on Location object properties.
3. What is document object model?
4. How do you assign object properties?
5. How do you create a new object in JavaScript?
6. What is the function of NaN?
7. What is the function of document object model?
8. Define screen object properties.
9. Discuss the methods of navigator object.
10. How to change font and color of a text element?

Answers: Self Assessment

1. top-level
2. Atob
3. setTimeout
4. Navigator objects
5. appVersion
6. javaEnabled
7. History Object
8. go(relPos | string)
9. length
10. location object
11. Hash
12. Host
13. Document object
14. bgColor
15. fgColor

10.9 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://www.comptechdoc.org/independent/web/cgi/javamanual/javadocument.html

http://www.w3resource.com/javascript/client-object-property-method/window.php

Unit 11: Handling Events Using JavaScript

CONTENTS

Objectives

After studying this unit, you will be able to:

- Explain the working of event
- Describe different events in JavaScript

Introduction

By using JavaScript, we have the ability to create dynamic web pages. Events are actions that can be detected by JavaScript.

Every element on a web page has certain events which can trigger a JavaScript. For example, we can use the onClick event of a button element to indicate that a function will run when a user clicks on the button. We define the events in the HTML tags.

Examples of events:

- A mouse click
- A web page or an image loading
- Mousing over a hot spot on the web page
- Selecting an input field in an HTML form
- Submitting an HTML form
- A keystroke

11.1 Working on Event

Events are normally used in combination with functions, and the function will not be executed before the event occurs!

11.1.1 onLoad and onUnload

The onLoad and onUnload events are triggered when the user enters or leaves the page.

The onLoad event is often used to check the visitor's browser type and browser version, and load the proper version of the web page based on the information.

Both the onLoad and onUnload events are also often used to deal with cookies that should be set when a user enters or leaves a page.

Example: You could have a popup asking for the user's name upon his first arrival to your page. The name is then stored in a cookie. Next time the visitor arrives at your page, you could have another popup saying something like: "Welcome upendra!".

11.1.2 onFocus, onBlur and onChange

The onFocus, onBlur and onChange events are often used in combination with validation of form fields.

Below is an example of how to use the onChange event. The checkEmail() function will be called whenever the user changes the content of the field:

```
<input type="text" size="30" id="email" onchange="checkEmail()">
```

11.1.3 onSubmit

The onSubmit event is used to validate ALL form fields before submitting it.

Below is an example of how to use the onSubmit event. The checkForm() function will be called when the user clicks the submit button in the form. If the field values are not accepted, the submit should be cancelled. The function checkForm() returns either true or false. If it returns true the form will be submitted, otherwise the submit will be cancelled:

```
<form method="post" action="xxx.htm" onsubmit="return checkForm()">
```

11.1.4 onMouseOver and onMouseOut

onMouseOver and onMouseOut are often used to create "animated" buttons.

Below is an example of an onMouseOver event. An alert box appears when an onMouseOver event is detected:

```
<a href="a.html" onmouseover="alert('An onMouseOver event');return false
</a>
```

Self Assessment

Fill in the blanks:

1. .... are normally used in combination with functions, and the function will not be executed before the event occurs!
2. The .... and onUnload events are triggered when the user enters or leaves the page.

3. The .... event is often used to check the visitor's browser type and browser version, and load the proper version of the web page based on the information.
4. The onFocus, onBlur and onChange events are often used in combination with .... of form fields.
5. The .... event is used to validate ALL form fields before submitting it.
6. onMouseOver and onMouseOut are often used to create "............................" buttons.

11.2 onClick Event Handler and onSelect Event Handler

In this section, we will discuss onClick Event Handler and onSelect Event Handler.

11.2.1 onClick Event Handler

The **onclick** property returns the onClick event handler code on the current element.

Notes When using the onclick event to trigger an action, also consider adding this same action to the onkeydown event, to allow the use of that same action by people who don't use a mouse or a touch screen.

Syntax

```
element.onclick = functionRef;
```

where *functionRef* is a function - often a name of a function declared elsewhere or a *function expression*.

Did u know? The event object passed to the specified event handler function is a MouseEvent.

11.2.2 onSelect Event Handler

A select event occurs when a user selects some of the text within a text or textarea field.

Caution The onSelect event handler executes JavaScript code when a select event occurs.

Example: The following example uses an onSelect handler in the *valueField* text object to call the selectState() function.

```
<INPUT TYPE="text" VALUE="" NAME="valueField" onSelect="selectState()">
```

Self Assessment

Fill in the blanks:

7. The .... property returns the onClick event handler code on the current element.
8. .... is often a name of a function declared elsewhere or a function expression.

9. The event object passed to the specified event handler function is a ....
10. A .... event occurs when a user selects some of the text within a text or textarea field.
11. The .... handler executes JavaScript code when a select event occurs.

11.3 Summary

- By using JavaScript, we have the ability to create dynamic web pages.
- Events are normally used in combination with functions, and the function will not be executed before the event occurs!
- The onLoad and onUnload events are triggered when the user enters or leaves the page.
- The onFocus, onBlur and onChange events are often used in combination with validation of form fields.
- The onSubmit event is used to validate ALL form fields before submitting it.
- When using the onclick event to trigger an action, also consider adding this same action to the onkeydown event.
- The onclick property returns the onClick event handler code on the current element.
- The event object passed to the specified event handler function is a MouseEvent.

11.4 Keywords

Events: Events are normally used in combination with functions, and the function will not be executed before the event occurs.

MouseEvent: The event object passed to the specified event handler function is a MouseEvent.

onclick: The onclick property returns the onClick event handler code on the current element.

onLoad: This event is often used to check the visitor's browser type and browser version, and load the proper version of the web page based on the information.

onSelect: A select event occurs when a user selects some of the text within a text or textarea field.

onSubmit: This event is used to validate ALL form fields before submitting it.

11.5 Review Questions

1. What do you mean by events?
2. Why we use events in JavaScript?
3. How to use onMouseOver and onMouseOut events?
4. How to change the background color of an element?
5. How do you define onLoad and onUnload event?
6. Define onLoad calls and onUnload calls.
7. Explain onBlur Event Handler.
8. Why we use onSelect Event Handler?

Notes

9. Describe onSubmit Event Handler.
10. Make distinction between onMouseOver and onMouseOut.

Answers: Self Assessment

1.	Events	2.	onLoad
3.	onLoad	4.	Validation
5.	onSubmit	6.	Animated
7.	onclick	8.	functionRef
9.	MouseEvent	10.	Select
11.	onSelect event		

11.6 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://books.google.co.in/books?id=Yg9BXNlAwloC&pg=PA32&lpg=PA32&dq=dial+m+in+dom+model&source=bl&ots=X3Ac6Ftw5x&sig=4TIsRZz4-SgE6DT1WXPNRieZDd0&hl=en&sa=X&ei=alKBUbidFcysrAfA2YCYAw&ved=0CDAQ6AEwAg#v=onepage&q=dial%20m%20in%20dom%20model&f=false

http://www.w3schools.com/dom/dom_node.asp

Unit 12: HTML Forms

Notes

CONTENTS

Objectives

After studying this unit, you will be able to:

- Explain the properties and methods of HTML forms
- Discuss HTML Button
- Explain HTML Text
- Describe HTML Text Area
- Explain HTML Checkboxes
- Discuss the HTML Radio Button
- Explain in detail the HTML option and select element

Introduction

A form is exactly what it says it is, a form that the user fills-in. It can have a variety of different types of input fields that you specify when you design the form (e.g. name, address, age, etc.) and the user fills-in prior to submitting the form. The only problem with HTML forms is that they are not much use by themselves because a traditional HTML form will only work in conjunction with a specially written server-side CGI program or script.

12.1 Properties and Methods

The method attribute tells the server how to submit the form information. There are two methods, get and post. The default method is get.

```
method="get"
```

This sends the form information by including it on the URL. The get method sends the information to the server in one step. It is best used for forms where the information is not secure (as it can be seen in the URL), and the amount of information passed is small. Get requests are useful if you want to call a CGI from a link.

```
<form method="get" action="">
```

When to use the get method:

- Short forms, with only 1 or 2 input fields
- Forms with primarily select, radio, and checkbox fields
- When the form needs to be called from a link
- For mailto forms

```
method="post"
```

This sends the form information in the HTTP environment. It is a little more secure, as your customers can't see it directly (although it is sent in clear text, so it can be hijacked by hackers with packet sniffers).

Notes The post method sends the data to the server in two steps. First the browser contacts the server and after the contact is made, it sends the information.

It is best used when you have a lot of information to pass (either textarea tags, or just a lot of fields.)

```
<form method="post" action="">
```

When to use the post method:

- Longer forms, more than 3 input fields
- Forms with large textarea fields
- Forms where security is more important (but if security is important, you should use a secure server too)

Task Compare and contrast get method and post method.

12.1.1 HTML Form E-mail

Generally, the button should be the last item of your form and have its name attribute set to "Send" or "Submit".

In addition to adding the submit button, we must also add a destination for this information and specify how we want it to travel to that place. Adding the following attributes to your <form>will do just this.

Method: We will only be using the post functionality of method, which sends the data without displaying any of the information to the visitor.

Action: Specifies the URL to send the data to. We will be sending our information to a fake email address.

Self Assessment

Fill in the blanks:

1. .... method sends the form information by including it on the URL.
2. .... method sends the form information in the HTTP environment.
3. .... attribute specifies the URL to send the data to.

12.2 Button

We use the <input> tag to create a basic button. Within the code, we use type="button" to set the control to a button.

Example:

```
<button type="button" onclick="JavaScript:alert('You liked that, didn\'t
you!')">Go ahead... make my day!</button>
```

Self Assessment

Fill in the blanks:

4. We use the .... tag to create a basic button.
5. Within the code, we use type="................................" to set the control to a button.

12.3 Text

Text fields offer a small rectangular box that's always ready to receive information from viewers. Users will notice that when they click these fields, the cursor will change from the typical arrow to a pipe character (|), allowing for text entries to be typed inside each input field.

12.3.1 TYPE Attribute

A text field is placed on a web page using the <input> tag, with the *type* attribute set with a value of "text".

```
<form name="MyForm" action="to:youremail@email.com" method="post">
```

```
First: <input title=" Enter First Name" id="first" name="first" type="text"
/> Last: <input title="Please Enter Your Last Name" id="last" name="last"
type="text" />
<input type="submit" value="SUBMIT" />
</form>
```

Text fields are designed to capture single words or phrases from the user. That information may then be processed through some kind of client/server side script (PHP, PERL, JavaScript). If you do plan on processing the data, be sure to include the *name* and *id* attributes.

Did u know? A descriptive *title* is also a great visual aid for providing a tool-tip display for your web elements.

12.3.2 HTML - Text Links

The World Wide Web got its spidery name from the plentiful connections (links) that link websites together with the click of a button. What most people don't know is that HTML links are actually HTML anchors constructed using anchor tags (<a>).

Example:

```
<a>I am a text link! </a>
```

12.3.3 Font

Font face and color depends entirely on the computer and browser that is being used to view your page. But the <font> tag is used to add style, size, and color to the text on your site. You can use a <basefont> tag to set all of your text to the same size, face, and color.

The font tag is having three attributes called size, color, and face to customize your fonts.

To change any of the font attributes at any time within your page, simply use the <font> tag.

Caution The text that follows will remain changed until you close with the </font> tag.

You can change any or all of the font attributes at the one time, by including all the required changes within the one <font> tag.

Example:

```
<font size="1">Font size="1"</font>
<font size="2">Font size="2"</font>
<font size="3">Font size="3"</font>
<font size="4">Font size="4"</font>
<font size="5">Font size="5"</font>
<font size="6">Font size="6"</font>
<font size="7">Font size="7"</font>
```

12.3.4 Headers

The HTML <header> tag represents a group of introductory or navigational aids.

Headers can contain headings, subheadings, version information, navigational controls, etc.

The <header> element is intended to usually contain the section's heading (an <h1>-<h6> element or an <hgroup> element), but this is not required. The <header> element can also be used to wrap a section's table of contents, a search form, or any relevant logos.

Example:

```
<html>
<head></head>
<body>
<header>
<span style="color:brown;font-style:italic;">abc</span>
<hr>
<hgroup>
<h1>welcome </h1>
<h2>Humans aren't the only talkers!</h2>
</hgroup>
</header>
<article>
<p>Ever encountered a talking tree? </p>
</article>
<footer>
<hr>
Welcome1
</footer>
</body>
</html>
```

12.3.5 Commenting in the HTML Coding

Comments are a great asset to new developers and a great way to place little notes to yourself reminding yourself what pieces of code are doing what. Comments are also great ways to troubleshoot bugs and code errors, as they give you the ability to comment out lines of code one at a time to search for the exact line causing problems.

As a sprouting young web developer, HTML code comments are your friends! A comment is a way to control which lines of code are to be ignored by the web browser and which lines of code to incorporate into your web page. There are three main reasons why you may want your code to be commented out or ignored.

- Comment out elements temporarily rather than removing them, especially if they've been left unfinished.
- Write notes or reminders to yourself inside your actual HTML documents.
- Create notes for other scripting languages like JavaScript which requires them.

Comment Tags:

<!– Opening Comment Tag

–> Closing Comment Tag

Self Assessment

Fill in the blanks:

6. A text field is placed on a web page using the <input> tag, with the .... attribute set with a value of "text".
7. The HTML .... tag represents a group of introductory or navigational aids.

12.4 Text Area

Text areas are text fields that can span several lines.

Unlike most other form fields, text areas are not defined with an <input> tag.

Instead you enter a <textarea> tag where you want the text area to start and a closing </textarea> tag where you want the area to end.

Everything written between these tags will be presented in the text area box.

Example:

```
<html>
<head>
<title>My Page</title>
</head>
<body>
<form name="myform" action="http://www.mydomain.com/myformhandler.cgi"
method="POST">
<div align="center">
```

This is outside the area<br><br>

```
<textarea cols="40" rows="5" name="myname">
```

Now we are inside the area - which is nice.

```
</textarea>
<br><br>
```

And now we are outside the area again.

```
</div>
</form>
</body>
</html>
```

Self Assessment

Fill in the blanks:

8. .... are text fields that can span several lines.
9. We use .... tag to end the text area.

12.5 HTML Checkboxes

Check boxes are used when you want to let the visitor select one or more options from a set of alternatives. If only one option is to be selected at a time you should use radio buttons instead.

Notes

Example:

```
<form name="myForm" action="http://localhost:8080/abc.html" method="post">
<p>Please select every sport that you play.</p>
Soccer: <input type="checkbox" name="sports" value="soccer" /><br />
Football: <input type="checkbox" name="sports" value="football" /><br />
Baseball: <input type="checkbox" name="sports" value="baseball" /><br />
Basketball: <input type="checkbox" name="sports" value="basketball" />
</form>
```

Self Assessment

Fill in the blanks:

10. .... are used when you want to let the visitor select one or more options from a set of alternatives.
11. If only one option is to be selected at a time you should use ....

12.6 HTML Radio Buttons

The Radio Button in HTML are type of input form, which allows a user to select any one option from the alternative options.

Caution To achieve this, we must specify the name of radio button properly.

The code enables a user to select one radio button at a time among alternative buttons.

<input type>: The <input> type specify the component type is radio button.

<value>: The <value> specify the output value of input element in html page, when it is clicked.

Did u know? The Output result is sent to the specified form's action URL.

Example:

```
<form action="">
```

I have a bicycle:

```
<input type="radio" name="vehicle" value="bicycle">
```

I have a bike:

```
<input type="radio" name="vehicle" value="bike">
```

I have a car:

```
<input type="radio" name="vehicle" value="car">
</form>
```

Self Assessment

Fill in the blanks:

12. The Radio Button in HTML are type of .... form, which allows a user to select any one option from the alternative options.

13. The .... specify the output value of input element in html page, when it is clicked.

12.7 Select and Option Element

HTML *select* fields provide essentially the same functionality as HTML Checkbox Fields. They allow the user to select one or more values from a predetermined series of options.

Notes Incorporating a select field into a web page is done using the <select> tag.

List values are then added to the field using the <option> tag, similar to how list items <li> are added to ordered list elements (<ol>).

Example:

```
<select name="selectionField">
 <option value="CA" >California — CA </option>
 <option value="CO" >Colorado — CO</option>
 <option value="CN" >Connecticut — CN</option>
</select>
```

12.7.1 HTML Drop Down Lists

Drop-down menus are probably the most flexible objects you can add to your forms.

Depending on your settings, drop-down menus can serve the same purpose as radio buttons (one selection only) or check boxes (multiple selections allowed).

The advantage of a drop-down menu, compared to radio buttons or check boxes, is that it takes up less space.

But that is also a disadvantage, because people can’t see all options in the menu right away.

There is a workaround for this - with the size setting, you can customize the menu so it shows more than just one option at a time, but when you do that – you also lose the advantage of taking up less space.

Sometimes you may want to replace text fields with drop-down menus. This might be because selecting from a menu is easier than typing. But it could also be because the script that handles the form can’t interpret just any text entry.

Example: You will often be asked to choose your state from a drop-down menu. This might be because picking it from the menu is easier than typing the name of the state.

Along the same line, you may often asked to enter the 2 letter initials of your state from a drop-down menu as well.

This could prevent confusion for the script that handles the form input. If, say, the script was programmed to only accept capital letters, then a drop-down menu would secure that no invalid entries were made.

Example:

```
<html>
<head>
<title>My Page</title>
</head>
<body>
<form name="myform" action="http://www.mydomain.com/myformhandler.cgi"
method="POST">
<div align="center">
<select name="mydropdown">
<option value="Milk">Fresh Milk</option>
<option value="Cheese">Old Cheese</option>
<option value="Bread">Hot Bread</option>
</select>
</div>
</form>
</body>
</html>
```

Task Analyse the use of HTML Drop Down Lists.

Self Assessment

Fill in the blanks:

14. HTML ....fields provide essentially the same functionality as HTML Checkbox Fields.

15. Depending on your settings, ... can serve the same purpose as radio buttons or check boxes.

12.8 Summary

- A form is exactly what it says it is, a form that the user fills-in.
- The method attribute tells the server how to submit the form information.
- Text fields offer a small rectangular box that's always ready to receive information from viewers.
- A text field is placed on a web page using the <input> tag, with the *type* attribute set with a value of "text".
- The World Wide Web got its spidery name from the plentiful connections (links) that link websites together with the click of a button.
- Font face and color depends entirely on the computer and browser that is being used to view your page.
- The HTML <header> tag represents a group of introductory or navigational aids.
- Comments are a great asset to new developers and a great way to place little notes to yourself reminding yourself what pieces of code are doing what.

12.9 Keywords

<header> tag: This tag represents a group of introductory or navigational aids.

Check boxes: These are used when you want to let the visitor select one or more options from a set of alternatives.

Comments: Comments are a great asset to new developers and a great way to place little notes to yourself reminding yourself what pieces of code are doing what.

Drop down menus: Depending on your settings, drop-down menus can serve the same purpose as radio buttons (one selection only) or check boxes (multiple selections allowed).

get: The get method sends the form information by including it on the URL.

Method: The method attribute tells the server how to submit the form information.

post: Post method sends the form information in the HTTP environment.

Text area: These are text fields that can span several lines.

12.10 Review Questions

1. What are forms? How they are created?
2. How do you use scripts on HTML?
3. How do you use Form's Action Attribute and Submit Button in HTML?
4. How do you use Text Field and Radio Button in Form?
5. How do you understand about Forms output?
6. How to create tables in HTML?
7. Write some useful table tags that we use in HTML.
8. Discuss the significance of comments in HTML.
9. Illustrate how to use HTML Checkboxes in form. Give example.
10. Describe the use of <font> with example.

Answers: Self Assessment

1.	get	2.	Post
3.	Action	4.	<input>
5.	button	6.	Type
7.	<header>	8.	Text areas
9.	</textarea>	10.	Check boxes
11.	radio buttons	12.	Input
13.	<value>	14.	Select
15.	drop-down menus		

12.11 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://www.echoecho.com/

http://www.quackit.com/html_5/tags/html_header_tag.cfm

http://www.tizag.com/

http://www.tutorialspoint.com/html/html_fonts.htm

Unit 13: Building Object of JavaScript

CONTENTS

Objectives

After studying this unit, you will be able to:

- Using JavaScript as a scientific calculator
- Explain Math Object
- Explain String Object
- Explain Date Object
- User Defined Object

Introduction

Objects in JavaScript, just as many other programming languages, can be compared to objects in real life. The concept of objects in JavaScript can be understood with real life, tangible objects.

In JavaScript, an object is a standalone entity, with properties and type.

Example: Compare it with a cup. A cup is an object, with properties. A cup has a colour, a design, weight, a material it is made of, etc. The same way, JavaScript objects can have properties, which define their characteristics.

13.1 Using JavaScript as a Scientific Calculator

JavaScript's Math object provides advanced arithmetic and trigonometric functions, expanding on JavaScript's basic arithmetic operators (plus, minus, multiply, divide). The Math object in JavaScript is borrowed from Java. In fact, the implementation of the Math object in JavaScript closely parallels the Math class in Java, except that the JavaScript Math object offers fewer methods.

JavaScript's Math object properties are treated as constants. In fact, the property names are in all upper-case, following the usual convention of capitalizing variable constants. These properties return often-used values, including ***pi*** and the square root of 2. The Math methods are used in mathematical and trigonometric calculations.

Notes Handy Math-object methods include ceilings, floor,paw, exp (exponent), max, min, round, and random. (Random is only available when using the X Window platform, however.)

The Math object is static, so you don't need to create a new Math object in order to use it. To access the properties and method of the Math object, you merely specify the Math object, along with the method or property.

Example: To return the value of *pi*, you use:

var pi = Math. PI;

Similarly, to use a math method you provide the name of the method, along with the parameters to use.

Example: To round the value of *pi*, you'd use:

var pi = Math.PI;

var pieAreRound = Math.round(pi); // displays

JavaScript does not recognize the keywords PI and round all by themselves. Exception: you may use with statement to associate the names of methods and properties with the Math object.

Did u know? This technique is a handy space-saver when you must use several.

The previous example can be written as

```
With (Math)
{
var pi = PI; Notes
var pieAreRound = round(pi);
alert (pieAreRound)
}
```

13.1.1 JavaScript Objects for a Date

The Date object is useful when you want to display a date or use a timestamp in some sort of calculation. In Java, you can either make a Date object by supplying the date of your choice, or you can let JavaScript create a Date object based on your visitor's system clock. It is usually best to let JavaScript simply use the system clock.

Syntax

```
new Date( )
new Date(milliseconds)
new Date(datestring)
new Date(year,month,date[,hour,minute,second,millisecond ])
```

The Date object has been created, and now we have a variable that holds the current date! To get the information we need to print out, we have to utilize some or all of the following functions:

- ***getTime() -*** Number of milliseconds since 1/1/1970 @ 12:00 AM
- ***getSeconds() -*** Number of seconds (0-59)
- ***getMinutes() -*** Number of minutes (0-59)
- ***getHours() -*** Number of hours (0-23)
- ***getDay() -*** Day of the week(0-6). 0 = Sunday, ... , 6 = Saturday
- ***getDate() -*** Day of the month (0-31)
- ***getMonth() -*** Number of month (0-11)
- ***getFullYear() -*** The four digit year (1970-9999)

Example:

```
<h4>It is now
<script type="text/javascript">
<!—
var currentTime = new Date()
var month = currentTime.getMonth() + 1
var day = currentTime.getDate()
var year = currentTime.getFullYear()
document.write(month + "/" + day + "/" + year)
//—>
</script>
</h4>
```

13.1.2 Creating Arrays and Objects

JavaScript objects are collections of properties and methods. A method is a function that is a member of an object. A property is a value or set of values (in the form of an array or object) that is a member of an object. JavaScript supports four kinds of objects: intrinsic objects, objects you create, host objects, which are provided by the host (such as window and document in Internet Explorer) and ActiveX objects.

All objects in JavaScript support "expando" properties and methods, or properties that can be added and removed at run time. These properties and methods can have any name, including numbers. If the name of the property or method is a simple identifier, it can be written after the object name with a period, such as:

```
var myObj = new Object();
myObj.name = "Fred";
myObj.age = 42;
myObj.getAge =
 function () {
 return this.age;
 };

document.write(myObj.name);
document.write("<br/>");
document.write(myObj.age);
document.write("<br/>");
document.write(myObj.getAge());
```

If the name of the property or method is not a simple identifier, or it is not known at the time you write the script, you can use an expression inside square brackets to index the property.

Caution The names of all expando properties in JavaScript are converted to strings before being added to the object.

In JavaScript, objects and arrays are handled almost identically, because arrays are merely a special kind of Object. Both can have properties and methods.

Arrays have a length property, but objects do not. When you assign a value to an element of an array whose index is greater than its length (for example, myArray[100] = "hello"), the length property is automatically increased to the new length. Similarly, if you make the length property smaller, any element whose index is outside the length of the array is deleted.

Example:

```
/ An array with three elements
var myArray = new Array(3);

// Add some data
myArray[0] = "Hello";
myArray[1] = 42;
myArray[2] = new Date(2000, 1, 1);

document.write("original length is: " + myArray.length);
document.write("<br/>");
// Add some expando properties
myArray.expando = "JavaScript!";
myArray["another Expando"] = "Windows";

// This will still display 3, since the two expando properties
// don't affect the length.
document.write("new length is : " + myArray.length);
```

Self Assessment

Fill in the blanks:

1. JavaScript's .... object provides advanced arithmetic and trigonometric functions, expanding on JavaScript's basic arithmetic operators.

2. The .... object is useful when you want to display a date or use a timestamp in some sort of calculation.
3. A .... is a value or set of values (in the form of an array or object) that is a member of an object.
4. A .... is a function that is a member of an object.

13.2 String Object

The **String** object let's you work with a series of characters and wraps Javascript's string primitive data type with a number of helper methods.

Because Javascript automatically converts between string primitives and String objects, you can call any of the helper methods of the String object on a string primitive.

Syntax

Creating a **String** object:

The *string* parameter is series of characters that has been properly encoded.

String Properties:

Here is a list of each property and their description.

Property	Description
constructor	Returns a reference to the String function that created the object.
length	Returns the length of the string.
prototype	The prototype property allows you to add properties and methods to an object.

Task Illustrate the use of string object with example.

13.2.1 Chaining Methods

JavaScript provides all sorts of shortcut ways of specifying your code which makes the code quicker to write. One of these that relates to objects is the chaining together of two or more methods which need to be run on the same object. Instead of having to specify the object on the front of each method as we want to run it, we can chain all the methods together one after the other on the one object reference.

Caution We just need to define the methods the right way in order to allow it.

Let's look at an example of this to see how it works. Here's a section of the code where we define an object and then run several methods on the object.

```
var newB = new example();
newB.firstMethod('x');
newB.secondMethod('y');
newB.thirdMethod(1,15,'z');
```

As you can see from the code none of the values returned from any of out three methods are actually being used since none of them have the returned value assigned to anything. Quite probably the way you currently have these methods defined you don't have them returning anything at all. We are going to change that.

What we need to do with all these methods that do not currently return anything is to add one extra line of code to the end of each of the methods.

```
return this;
```

With that line of code in place our methods now each return the object that they belong to This is the only change that we need to make to methods in order to allow us to use method chaining. Of course we can't add that line if the method already returns a value but them methods that return values are not suited to method chaining in the first place as each of those needs to be specified separately in order to be able to assign the value that is returned.

With our methods that don't normally need to return a value now returning a reference to the object that they belong to it is no longer necessary for us to use three separate statements to call those three methods. We can now combine all three into one statement simply by chaining the methods together.

```
var newB = new example();
newB.firstMethod('x').secondMethod('y').thirdMethod(1,15,'z');
```

This works because the last statement in firstMethod() is **return this;** which in this case returns a reference to newB. This means that newB.firstMethod() after the method has run is equivalent to newB and so can be substituted for newB where we want the method to be run and the object to be available for further referencing.

Self Assessment

Fill in the blanks:

5. The .... object let's you work with a series of characters and wraps Javascript's string primitive data type with a number of helper methods.
6. The .... property allows you to add properties and methods to an object.

13.3 The Math Object

JavaScript provides all sorts of shortcut ways of specifying your code which makes the code quicker to write. Unlike the other global objects, *Math* is not a constructor. All properties and methods of Math are static and can be called by using *Math* as an object without creating it.

Thus, you refer to the constant pi as **Math.PI** and you call the *sine* function as **Math.sin(x)**, where x is the method's argument.

Properties	Description
E	The constant of E, the base of natural logarithms.
LN2	The natural logarithm of 2.
LOG2E	Base 2 logarithm of E.
LOG10	E Base 10 logarithm of E.
SQRT1_2	Square root of 1/2.

Self Assessment

Fill in the blanks:

7. Unlike the other global objects, *Math* is not a ....

8. All properties and methods of Math are .... and can be called by using *Math* as an object without creating it.

13.4 Date Object

The Date object is used to work with dates and times.

Date objects are created with the Date () constructor.

There are four ways of instantiating a date:

```
new Date() // current date and time
new Date(milliseconds) //milliseconds since 1970/01/01
new Date(dateString)
new Date(year, month, day, hours, minutes, seconds, milliseconds)
```

Most parameters above are optional. Not specifying causes 0 to be passed in.

Once a Date object is created, a number of methods allow you to operate on it. Most methods allow you to get and set the year, month, day, hour, minute, second, and milliseconds of the object, using either local time or UTC (universal, or GMT) time.

All dates are calculated in milliseconds from 01 January 1970 00:00:00 Universal Time (UTC) with a day containing 86,400,000 milliseconds.

Some examples of instantiating a date:

```
today = new Date()
d1 = new Date("October 13, 1975 11:13:00")
d2 = new Date(79,5,24)
d3 = new Date(79,5,24,11,33,0)
```

13.4.1 Using Date Methods

We can easily manipulate the date by using the methods available for the Date object.

In the example below we set a Date object to a specific date (14th January 2010):

```
var myDate=new Date();
myDate.setFullYear(2010,0,14);
```

And in the following example we set a Date object to be 5 days into the future:

```
var myDate=new Date();
myDate.setDate(myDate.getDate()+5);
```

Notes If adding five days to a date shifts the month or year, the Date object itself handles the changes automatically!

Self Assessment

Fill in the blanks:

9. The .... is used to work with dates and times.

10. Date objects are created with the .... constructor.

13.5 User Defined Objects

All user-defined objects and built-in objects are descendants of an object called Object.

13.5.1 The New Operator

The new operator is used to create an instance of an object. To create an object, the new operator is followed by the constructor method.

In the following example, the constructor methods are Object(), Array(), and Date().

These constructors are built-in JavaScript functions.

```
var employee = new Object();
var books = new Array("C++", "Perl", "Java");
var day = new Date("August 15, 1947");
```

13.5.2 The Object () Constructor

A constructor is a function that creates and initializes an object. JavaScript provides a special

constructor function called Object () to build the object. The return value of the Object () constructor is assigned to a variable.

The variable contains a reference to the new object.

Did u know? The properties assigned to the object are not variables and are not defined with the var keyword.

This example demonstrates how to create an object:

```
<html>
<head>
<title>User-defined objects</title>
<script type="text/javascript">
var book = new Object(); // Create the object
book.subject = "Perl"; // Assign properties to the object
book.author = "Mohtashim";
</script>
</head>
<body>
<script type="text/javascript">
```

Notes

```
document.write("Book name is : " + book.subject + "<br>");
document.write("Book author is : " + book.author + "<br>");
</script>
</body>
</html>
```

Task Give examples of Math object and Date object.

Self Assessment

Fill in the blanks:

11. All user-defined objects and built-in objects are descendants of an object called ....
12. The .... operator is used to create an instance of an object.
13. A .... is a function that creates and initializes an object.
14. The .... contains a reference to the new object.
15. The properties assigned to the object are not variables and are not defined with the .... keyword.

13.6 Summary

- In JavaScript, an object is a standalone entity, with properties and type.
- JavaScript's Math object properties are treated as constants.
- The Date object is useful when you want to display a date or use a timestamp in some sort of calculation.
- JavaScript objects are collections of properties and methods. A method is a function that is a member of an object.
- A method is a function that is a member of an object.
- All objects in JavaScript support "expando" properties and methods, or properties that can be added and removed at run time.
- JavaScript provides all sorts of shortcut ways of specifying your code which makes the code quicker to write.
- JavaScript provides all sorts of shortcut ways of specifying your code which makes the code quicker to write.

13.7 Keywords

charAt(): This is a method returns the character at the index specified.

getHours(): Returns the hour

getMinutes(): Returns the minutes

getSeconds(): Returns the seconds

indexOf(): This method can be used to search down a string (from left to right) until it finds a string fragment matching the specified value.

Javascripts objects: JavaScript objects are collections of properties and methods.

Length: Length is a property of a string and receives its value indirectly based on the number of characters in a string.

Math object: JavaScript's Math object provides advanced arithmetic and trigonometric functions, expanding on JavaScript's basic arithmetic operators.

13.8 Review Questions

1. Explain, how JavaScript used as a scientific calculator.
2. What are the objects in JavaScript?
3. Define the methods of math object in JavaScript.
4. Discuss the properties of math object used in JavaScript.
5. How do you create a new object in JavaScript?
6. What are the user defined objects?
7. Explain the property and methods of date object.
8. What is string object?
9. Explain the Get methods of date object.
10. Differentiate between indexOf and lastindexOf methods in string object.

Answers: Self Assessment

1.	Math	2.	Date
3.	property	4.	Method
5.	String	6.	Prototype
7.	constructor	8.	Static
9.	Date object	10.	Date ()
11.	Object	12.	New
13.	constructor	14.	Variable
15.	var		

13.9 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Notes

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://javascript.about.com/od/objectorientedjavascript/a/oop20.htm

http://www.javascriptkit.com/javatutors/math.shtml

https://developer.mozilla.org/en-US/docs/JavaScript/Guide/Working_with_Objects

Unit 14: DHTML

CONTENTS

Objectives

After studying this unit, you will be able to:

- Discuss DHTML basic
- Describe in detail the Cascading style sheet
- Explain the class
- Describe External Style Sheets

Introduction

DHTML is based on thinking of a web page like a printed page: a document that is rendered once and that is static once rendered. The idea behind Dynamic HTML (DHTML), however, is to make every element of a page interactively controllable, before, during, and after the page is rendered. This means you can make things move, appear and disappear, overlap, change styles, and interact with the user to your heart's content. Through DHTML, users get a more engaging and interactive web experience without constant calls to a web server or the overhead of loading new pages, plug-ins, or large applets.

Notes

14.1 DHTML Basic

DHTML is not a language itself, but rather a combination of:

- HTML 4.0 (or XHTML 1.0)
- JavaScript — the Web's standard scripting language
- Cascading Style Sheets (CSS) — styles dictated outside a document's content
- Document Object Model (DOM) — a means of accessing a document's individual elements

Dynamic HTML (DHTML) is a set of innovative features originally introduced in Microsoft Internet Explorer 4.0. By enabling authors to dynamically change the rendering and content of a Web page as the user interacts with it, DHTML enables authors to create visually compelling Web sites without the overhead of server-side programs or complicated sets of controls to achieve special effects.

With DHTML, you can easily add effects to your pages that previously were difficult to achieve.

Example: you can:

- Hide content until a given time elapses or the user interacts with the page.
- Animate text and images in your document, independently moving each element from any starting point to any ending point, following a predetermined path or one chosen by the user.
- Embed a ticker that automatically refreshes its content with the latest news, stock quotes, or other data.
- Use a form to capture user input, and then instantly process and respond to that data.

DHTML achieves these effects by modifying the in-memory representation of the current document and automatically reformatting it to show changes. It does not reload the document, load a new document, or require a distant server to generate new content. Instead, it uses the user's computer to calculate and carry out changes. This means a user does not wait for text and data to complete time-consuming round trips to and from a server before seeing the results. Furthermore, DHTML does not require additional support from applications or embedded controls to make changes. Typically, DHTML documents are self-contained, using styles and a script to process user input and directly manipulate the HTML elements, attributes, styles, and text of the document.

Notes In short, DHTML eliminates the shortcomings of static pages. You can create innovative Web sites, on the Internet or on an intranet, without having to sacrifice performance for interactivity. Not only does DHTML enhance the user's perception of your documents, it also improves server performance by reducing requests to the server.

14.1.1 DHTML Technologies

Speaking in true sense there is nothing dynamic in DHTML but enclosing technologies such as CSS, JavaScript, DOM and the static markup language it becomes dynamic.

JavaScript: Whether we call it JavaScript, Jscript, or ECMAScript, it is the most common language used today for client-side scripting. The main reason for this JavaScript comes with virtually every browser.

Notes

Example: An onload event could execute a JavaScript function to query the browser's cookies collection to determine whether the user is a first-time visitor to the page.

CSS: It stands for Cascading Style Sheet. This is used for the presentation part of the web page. In simple words it holds the designing of the page. The look & feel of the page completely depends on CSS. In DHTML CSS rules can be modified at both the document and the element level using JavaScript with event handlers, they can add a significant amount of dynamism with very little code.

DOM: It stands for Dynamic Object Model and it is the weakest link in DHTML as many of the browser does not support the DOM functionality. It defines the object and its properties.

Did u know? It is a standard way of accessing and manipulating the static content.

The Document Object Model is a platform and language-neutral interface that allows program and scripts to dynamically access the content and update it.

Self Assessment

Fill in the blanks:

1. .... enables authors to create visually compelling Web sites without the overhead of server-side programs or complicated sets of controls to achieve special effects.
2. .... is used for the presentation part of the web page.
3. .... is the weakest link in DHTML as many of the browser does not support the DOM functionality.

14.2 Cascading Style Sheet

CSS was first developed in 1997, as a way for Web developers to define the look and feel of their Web pages. It was intended to allow developers to separate content from design so that HTML could perform more of the function that it was originally based on - the markup of con CSS is one of the most powerful tools a Web designer can learn because with it you can affect the entire mood and tone of a Web site. Well written style sheets can be updated quickly and allow sites to change what is prioritized or valued without any changes to the underlying XHTML.

There are essentially two parts to Cascading Style Sheets:

1. the styles
2. the stylesheet

What this means is that when you want to learn CSS, you need to learn both the style properties — the styles; and the placement of those properties — the stylesheet.

There are different types of cascading style sheets:

- ***External style sheet,*** which you use when you want to apply the same styles consistently across all the pages in your Web site that are linked to it.

Did u know? It is also known as linked style sheet.

- ***Embedded style sheet,*** which you use when you want to define styles for the current page.

Task Compare and contrast External style sheet and Embedded style sheet.

Self Assessment

Fill in the blanks:

4. CSS was first developed in ...., as a way for Web developers to define the look and feel of their Web pages.

5. External style sheet is also known as ....

6. .... style sheet is used when you want to define styles for the current page.

14.3 Class

You may be wondering if it is possible to give an HTML element multiple looks with CSS.

Example: Sometimes you want the font to be large and white, while other times you would prefer the font to be small and black. CSS would not be very useful if it did not allow you to have many different types of formats for a single HTML tag. Well, you are in luck! CSS allows you to do just that with the use of classes.

Using classes is simple. You just need to add an extension to the typical CSS code and make sure you specify this extension in your HTML.

Example: Making two paragraphs that behave differently. First, we begin with the CSS code, note the red text.

CSS Code:

```
p.first{ color: blue; }
p.second{ color: red; }
```

HTML Code:

```
<html>
<body>
<p>This is a normal paragraph.</p>
<p class="first">This is a paragraph that uses the p.first CSS code!</p>
<p class="second">This is a paragraph that uses the p.second CSS code!</p>
...
```

Self Assessment

State true or false:

7. CSS would not be very useful if it did not allow you to have many different types of formats for a single HTML tag.

8. For using classes, you are not required to add an extension to the typical CSS code.

Notes

14.4 External Style Sheets

An external style sheet is a separate file where you can declare all the styles that you want to use on your website. You then link to the external style sheet from all your HTML pages. This means you only need to set the styles for each element once.

Caution If you want to update the style of your website, you only need to do it in one place.

Example:

- Type the following into a plain text file, and save with a .css extension (i.e. external_style_sheet.css).

```
p {font-family: georgia, serif; font-size: x-small;}
hr {color: #000000; height: 1px }
a:hover {color: #ff0000; text-decoration: none}
```

- Add the following code between the <head></head> tags of all HTML documents that you want to reference the external style sheet. This code uses the HTML <link> element to link to the external style sheet.

```
<link rel=StyleSheet href="external_style_sheet.css" type="text/
css">
```

  Now, all of your HTML documents will use the styles from your external style sheet resulting in a consistent look and feel. If you want to change anything, you only need to update the external style sheet.

14.4.1 Using <SPAN> ... </SPAN> Tag

The SPAN element has very similar properties to the DIV element, in that it changes the style of the text it encloses. But without any style attributes, the SPAN element won't change the enclosed items at all.

The primary difference between the SPAN and DIV elements is that SPAN doesn't do any formatting of it's own. The DIV element includes a paragraph break. The SPAN element simply tells the browser to apply the style rules to whatever is within the SPAN.

To use the SPAN element, simply surround the text that you want to add styles to with the <span> and </span> tags:

```
<div id="mydiv">
<p><span class="highlight">Highlighted text</span> and non-highlighted text.</
p>
</div>
```

The SPAN element has no required attributes, but the three that are the most useful are the same as for the DIV element:

- style
- class
- id

Use SPAN when you want to change the style of elements without placing them in a new block-level element in the document.

Notes

Example: If you had a Level 3 Heading (H3) that you wanted the second word to be red, you could surround that word with <span style="color : #f00;">2ndWord</span> and it would still be a part of the H3 tag, just red. For example:

```
<h3>This is My <span style="color: red;">Awesome</span> Headline</h3>
```

14.4.2 Using <DIV> ... </DTV> Tag

The DIV elements defines logical divisions in your web page. It acts a lot like a P element, by placing newlines before and after the division. A division can have multiple paragraphs in it.

Using the DIV Tag

To use the DIV element, simply surround the area of your page that you want as a separate division with the <div> and </div> tags:

```
<div id="mydiv">
<p>contents of div</p>
</div>
```

The DIV element gives you the chance to define the style of whole sections of the HTML. You can define a division of your page as a call out and give that area a different style from the surrounding text. That area could have images, paragraphs, headlines, anything you wanted.

Notes The DIV element also gives you the ability to ID areas of your documents so that you can change them with Ajax and dynamic HTML.

The DIV element is different from the new HTML5 SECTION element because it does not give the enclosed content any semantic meaning. If you aren't sure whether the block of content should be a DIV or a SECTION, think about what that content's purpose is and why you need the DIV or SECTTION element.

- If you need the element simply to add styles to that area of the page, you should use the DIV element.
- If that area of the page has a specific meaning, for example it holds all your social media elements or it contains your blogroll, then you should use the SECTION element.

One thing to keep in mind when using the DIV element is that it breaks paragraphs. It acts as a paragraph end/beginning, and while you can have paragraphs within a DIV you can't have a DIV inside a paragraph.

The primary attributes of the DIV element are:

- style
- class
- id

Task Illustrate the use of div tag with example.

Self Assessment

Fill in the blanks:

9. An .... style sheet is a separate file where you can declare all the styles that you want to use on your website.
10. The .... element changes the style of the text it encloses.
11. The .... element includes a paragraph break.
12. The DIV elements defines .... divisions in your web page.
13. The DIV element gives you the chance to define the .... of whole sections of the HTML.

14.5 Advantages and Disadvantages of DHTML

In this section, we will discuss the advantages and disadvantages of DHTML.

14.5.1 Advantages of DHTML

Advantages of DHTML are as follows:

Small file sizes

DHTML files are small compared to other interactive media such as Flash or Shockwave. Therefore they have a shorter download time and take up less bandwidth.

Supported by both major browser manufacturers

Both Microsoft and Netscape currently support DHTML in some shape or form.

DHTML will be a standard

The World Wide Web Consortium or the W3C is currently implementing standards for DHTML technologies. It has already released preliminary specifications for DOM and CSS. These specifications lay the groundwork for more complete standards to come, which both Netscape and Microsoft have pledged to support.

No plug-ins necessary

Plug-ins are not needed to view DHTML files. A visitor to your site needs only a Netscape 4.0 browser or an Internet Explorer 4.0 browser. This puts fewer requirements on your audience;

Caution They don't need to download special software to view your site.

Doesn't require a Java Virtual Machine (JVM)

DHTML isn't a Java technology. DHTML provides many functions that can otherwise be attained through Java—a compiled, object-oriented computer language. Pages that contain Java applets require the user to wait for the JVM to start and for Java byte code to download, which takes quite a bit of time and bandwidth. Although Java is good for some applications, DHTML can be an attractive alternative for animations, design issues, and simple tasks.

14.5.2 Disadvantages of DHTML

Disadvantages of DHTML are as follows:

- It can create only static and plain pages so if we need dynamic pages then HTML is not useful.

Notes

- Need to write lot of code for making simple webpage.
- Security features are not good in HTML.
- If we need to write long code for making a webpage then it produces some complexity.

Self Assessment

State true or false:

14. DHTML files are huge compared to other interactive media.
15. The World Wide Web Consortium or the W3C is currently implementing standards for DHTML technologies.

14.6 Summary

- Dynamic HTML (DHTML) is a set of innovative features originally introduced in Microsoft Internet Explorer 4.0.
- With DHTML, you can easily add effects to your pages that previously were difficult to achieve.
- Typically, DHTML documents are self-contained, using styles and a script to process user input and directly manipulate the HTML elements, attributes, styles, and text of the document.
- DHTML eliminates the shortcomings of static pages.
- ***DOM:*** It stands for Dynamic Object Model and it is the weakest link in DHTML as many of the browser does not support the DOM functionality.
- CSS was first developed in 1997, as a way for Web developers to define the look and feel of their Web pages.
- An external style sheet is a separate file where you can declare all the styles that you want to use on your website.
- Embedded style sheet is used when you want to define styles for the current page.

14.7 Keywords

CSS: It stands for Cascading Style Sheet. This is used for the presentation part of the web page. In simple words it holds the designing of the page.

DHTML: Dynamic HTML (DHTML) is a set of innovative features originally introduced in Microsoft Internet Explorer 4.0.

DIV: This element gives you the chance to define the style of whole sections of the HTML.

DOM: It stands for Dynamic Object Model and it is the weakest link in DHTML as many of the browser does not support the DOM functionality.

Embedded style sheet: Embedded style sheet is used when you want to define styles for the current page.

External Style Sheet: It is a separate file where you can declare all the styles that you want to use on your website.

SPAN element: The SPAN element changes the style of the text it encloses.

14.8 Review Questions

1. Differentiate between HTML and DHTML.
2. What are the advantages of DHTML over HTML?
3. What is the advantage of using an external style sheet?
4. What is CSS? Discuss.
5. What is class? Discuss.
6. What is external Style Sheet? How to link?
7. What are the features of DHTML?
8. How does DHTML work with JavaScript?
9. What are the attributes that make up a DHTML?
10. Illustrate the concept of embedded style sheet with example.

Answers: Self Assessment

1.	DHTML	2.	CSS (Cascading Style Sheet)
3.	DOM (Dynamic Object Model)	4.	1997
5.	linked style sheet	6.	Embedded
7.	True	8.	False
9.	external	10.	SPAN
11.	DIV	12.	Logical
13.	style	14.	False
15.	True		

14.9 Further Readings

Books

Hall, 2009, *Core Web Programming,* 2/E, Pearson Education India.

Jon Duckett, 2011, *Beginning Web Programming with HTML, XHTML and CSS,* John Wiley & Sons.

Robert F. Breedlove,1996, *Web Programming Unleashed,* Sams.net.

Tim Downey, 2012, *Guide to Web Development with Java: Understanding Website Creation,* Springer.

Online links

http://docstore.mik.ua/orelly/web2/wdesign/ch29_01.htm

http://msdn.microsoft.com/en-us/library/ms533044%28v=vs.85%29.aspx

http://office.microsoft.com/en-in/training/introduction-to-cascading-style-sheets-css-RZ001211122.aspx?section=5

Notes

http://webdesign.about.com/od/css/a/aa010702a.htm

http://webdesign.about.com/od/htmltags/a/aa011000a.htm

http://www.careerride.com/dhtml-technologies.aspx

http://www.tizag.com/cssT/class.php

LOVELY PROFESSIONAL UNIVERSITY
Jalandhar-Delhi G.T. Road (NH-1)
Phagwara, Punjab (India)-144411
For Enquiry: +91-1824-521360
Fax.: +91-1824-506111
Email: odl@lpu.co.in

ISBN 978-93-87034-78-5
9 789387 034785