WEB TECHNOLOGY

SAMRAT KRISHNA GADDAM
DR.T.S.RAVI KIRAN,DR.A.SRISAILA

TO MY PARENTS AND BROTHER

for raising me to believe that anything was possible.

Foreword

This text book provides students with a comprehensible introduction to the programming and scripting languages currently used to create Web sites and Web applications—the main aim being to teach the programming concepts of various Web technologies and the fundamentals needed to program on the Internet. The book emphasises the underlying fundamentals of Web page development and prepares students to build real-world, industrial strength Web-based applications, and use a wide variety of Web development tools effectively and efficiently. Students are introduced to the concepts of Introduction to Web, HTML, JavaScript, XML, PHP & MySQL. The material presented on HTML contains an elaborate description with examples to help the reader clearly understand the networking concepts. The book is intended as a text for students of Computer Science and Engineering, Information Technology, Master of Computer Applications and M.Sc(Computer Science).

Preface

As this is an introductory level text, the goal is not to exhaust the covered topics. Instead we will focus on familiarizing ourselves with each language's abilities and how to fuse those languages and methods together to create a responsive, well-developed site. You will frequently find links and keywords throughout the text, which you can use as a starting point to further research topics that interest you. We will examine web development in a detailed manner with examples. The topics included below all lend themselves towards this goal by touching each aspect of the process.Unit-1 explains the concept of web where the readers will be able to get a brief idea about how the web works and the concept behind the web.Unit-2 coevrs the indepth concepts of HTML explaining the use of each and every tag with example programs.Unit-3 covers the scripting language part where JavaScript is explained which is used as an application of client side scripting.Chatpter-4 is dealt

with the XML which is a crutial part of Web Develeopment where the XML programs are explained in detail.Chapter-5 explain about how we will be able to connect a webpage to database using MySQL.

Acknowledgements

First and foremost, we are grateful to all the contributing authors Dr.T.S.Ravi Kiran and Dr.A.Srisaila for their time, effort, and understanding during the preparation of the book. We thank editors of Notion Press book series on Web Technologies, for their enthusiastic support and guidance during the preparation of book and enabling us to easily navigate through Notion Press publication process. All chapters were reviewed and authors have updated their chapters. We thank members for their time and effort in peer reviewing of chapters. Samrat Krishna would like to thank his family members, especially. Finally, we would like to thank the staff at Notion Press(Chennai). They were wonderful to work with!

Prologue

Web development is an evolving amalgamation of languages that work in concert to receive, modify, and deliver information between parties using the Internet as a mechanism of delivery.
While it is easy to describe conceptually, implementation is accompanied by an overwhelming variety of languages, platforms, templates, frameworks, guidelines, and standards. Navigating a project from concept to completion often requires more than mastery of one or two complementing languages, meaning today's developers need both breadth, and depth, of knowledge to be effective.

This textbook provides the reader with an understanding of the various elements of web development by focusing on the concepts and fundamentals through the examples within, providing a foundation that allows easier transition to other languages and a better understanding of how to approach their work.

CHAPTER ONE

Introduction to Web

1.1 Introduction

- Internet refers to a wide network through which computers are interconnected globally with one another.

OR

- Internet refers to millions of computers connected a gigantic network which communicate via TCP/IP protocols (Transmission Control Protocol/Internet Protocol).
- The collection of computers to store large amount of information.
- Internet is the biggest network in the world and freely accessible to the public.

1.2 History of the Internet:

- In 1957, the department of defence (military) in U.S government was developed one of the network i.e. ARPANET (Advanced Research Project Agency) for the areas of mathematics, pure sciences and engineering.
- In 1961, Leonard Kleinrock developed one of the networks i.e. called packet switching instead of streaming of data in wave forms. Packet switching sends data in packets.
- Packets in packet switching system are small and carry information on where the data becomes in the original data stream. Error checking ensures when data corrupted (i.e. if a packet is cost (or) corrupted, the receiving computer can request only the missing packet can be resent, not the entire data stream.
- In 1967, Wesley Clark developed one of the network i.e. IMP (Interface Message Processor). IMP provides interface between a computer and the wide area network (WAN) i.e. it allows different types of computers to talk to each other.
- In 1972, Bob Kahn proposed the idea of an open architecture for the internet. The idea was to allow any computer to plug into the internet and be able to function with no modification to the computer itself.
- In 1973, Bob Kahn and Vinton Cerf working on a detailed protocol (TCP/IP) that addressed Kahn's internet working standards.
- TCP/IP protocol divides the process of transmition data into 2 steps

a. TCP handles the data (transmition) and correction of the stream transmission errors.

b. IP handles the building of data into packets, addressing them and handles the errors where the data needs to be resent.

- Each computer on the internet had a unique IP address, which has 4 parts. Part being between 0 to 255. This system supports only 4 billion possible combinations.

- With the increase in the number of computers connected to the internet in the 70's and 80's Paul Mockapetris developed one of the system is called Domain Name System (DNS).
- DNS is a distributed mechanism for domain name resolution, where difference portions of the net handled only parts of the addresses.

1.3 Internet services and accessibility:

Today internet provides large number of services. Some of the services are:

- ***Email***: This service provide to send and receive the messages and to attach files.
- ***E-mailing lists:*** Everyone subscribed to the list gets a message sent to the list.
- ***USENET news groups:*** Electronic bulletin board service
- ***Real-time communications:*** Chat, messengers, tele-conferencing, video-conferencing.
- ***File transfer Protocol (FTP):*** A service that helps to send and receive files to and from a file server.
- ***Telnet:*** A remote login to other computers on the internet possibly anonymously.
- ***World Wide Web (WWW):*** Documents and files of various types which are connected using hypertext links to creates a web-link structure and are accessed through the internet by addresses called URLs. Users access the above services through internet service provider (ISP). Accessibility means how to provide internet connections. There are different types of internet connections. Those are:
- ***Dial up connections:*** To use this access, the computer should have a modem to connect to the phone system and software that uses the modem. This software instructs the modem (convert analog to digital, digital to analog) to place a telephone call to the number provided by the ISP. This is very popular because it is less expensive.
- ***High speed connections:*** It allows information to travel quickly. There are 4 types of high speed connections.

- ***Digital subscriber lines (DSL):*** In this one, the computer is always connected to the internet. This technology sends digital data through the existing phone lines to carry internet services.
- ***Cable:*** Use some wires (Fiberoptic, Coaxial) send the information on the internet.
- ***Satellite:*** Without using wire vase connection, we send data at high speed from the satellite.
- ***Integrated service digital network (ISDN):*** In this one we send voice, video and text data over telephone lines.

1.4 Uses of the internet:

- Telecommunicating (working from home or anywhere) and online conferencing.
- Business, advertising and online shopping.
- Jobs, news, online courses, coaching's.
- Government services, politics and national defence.
- Electronic publishing (magazines, news papers and news services)
- Entertainment (television, radio, video, audio, etc)
- Teaching and learning (course websites, conferencing, etc)
- General information about subject (using some searching web sites)
- Correspondence (email, chatting, etc)

1.5 Protocols:

Protocol is a set of rules follow the computers on a network use for communication with other computers i.e. they give specification on how the computers talk with each other (how the data sent). There are various protocols available. Those are:

- ***Ethernet:*** This is used to transfer information on a LAN and it follows physical layer (send data into machine (0, 1) format) to establish connection.
- ***Internet Protocol (IP):*** It provides a unique address to each computer.
- ***Transport Control Protocols (TCP):*** It breaks messages (large) transport them reliable and reassemble them.
- ***File Transfer Protocol (FTP):*** It is used to connect two computers over the internet and transfer files from one system to another.

- ***Hypertext Transport Protocol (HTTP):*** It is used to retrieve web pages from a web server.
- ***Simple Mail Transfer Protocol (SMTP):*** It is used for email transmissions.

1.6 Web concepts:

- The World Wide Web (www) is an international hypertext system that links together millions of documents.
- Hypertext Markup Language (HTML) is a collection of tags, which is used to create a formatted hypertext documents (i.e. develop any web pages).
- Website is a collection of web pages.
- A web browser is a program that displays the web pages it retrieves.

Ex: Microsoft Internet Explorer, Netscape Navigator, Mozilla Firefox, etc.

- ***Client/server model of the web:***
- Most internet services are follow on the client/server model.
- The internet user is the client and use client software installed on his computer to access various internet services.
- When a user wants to connect to particular information, he users his client software (internet explorer) to connect to server programs, which provide the information needed.

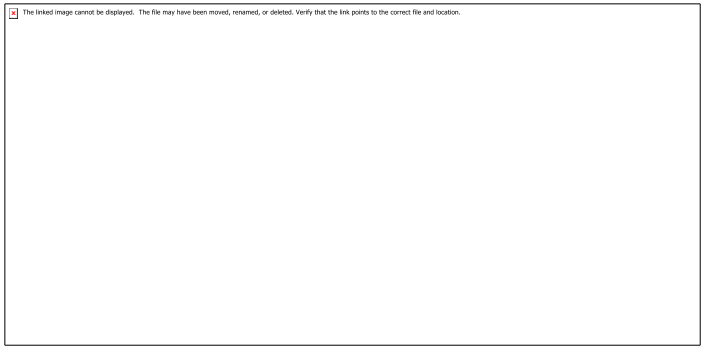


Fig 1.6.1 Client-Server model

Whenever the client connect to server they use me of the protocol i.e. HTTP.

- ***Retrieving data from the web:***

- We can retrieve the data from server by using address of the file i.e. called Uniform Resource Locator (URL).
- The address bar of the browser shows URL of the current displayed document.

protocol

- The URL consists of 3 parts domain name

Path

- The protocol is a set of rules follow connect one computer to another

  -
    - Ex: http://
- The domain name is the internet address of the computer (server) (website address). It may be expressed as IP address.
- The path is the directory and file specifications i.e. it specifies the computer to know which directory and file to access after connecting to the server.

Ex: http://www.w3schools.com/index.html

Protocol domain name path

- ***Web Browsers: Navigation features:***

Some of the default features available in the browser are:

- Back button returns you to the page you lost visited.
- Forward button acts reverse of the back button.
- Home button takes you back to a page that your browser has chosen as the home page.
- Reload (or) Refresh button redisplays a page that has been changed (or) did not display properly.
- Stop button stops loading (or) downloading a page.
- History button lists all the pages you have recently visited.
- Bookmarks (or) favorites store addresses so you can easily return to then later.

Searching information on the web:

-
  - If we know a website address you can type it in the address box and go there directly.
  - Today several tools are available to quickly search multiple sites on the internet.

a. A subject guide (or) directory is a website that organizes gets context hierarchically by subject (topics and sub topics).

b. A search engine searches the web for one (or) more keywords you type and displays a list of pages found.

c. A metasearch site submits the search to more than one search engine.

d. Using a human search service, we can hire area person to perform our search.

Internet standards:

Internet follow some standards whenever creation, testing and implementation data.

The Internet Engineering Task Force (IETF) develops these standards, next these are considered by Internet Engineering Steering Group (IESG), which appeal to the Internet Architecture Board (IAB).

The RFC editor is responsible for preparing and organizing these standards in their final form.

All the internet standards are given a number in the STD series.

Ex: STD1 (RFC3700) –Internet official protocol standards.

STD5 (RFC0791) –Internet Protocol (IP)

STD5 (RFC0792) –Internet Control Message Protocol

STD6 (RFC0768) –User Datagram Protocol (UDP)

STD7 (RFC0793) –Transmission Control Protocol (TCP)

STD9 (RFC0959) –File Transfer Protocol

STD13 (RFC1034) –Domain Names Concepts and Facilities

STD13 (RFC1035) –Domain names Implementation and specification

STD62 (RFC3413) –Simple network messages protocol (SNMP)

STD66 (RFC3986) –Uniform Resource Identifier (URI)

CHAPTER TWO

HTML

2.1 Introduction to HTML:

HTML means Hypertext Markup Language. In 1960 Ted Nelson introduced Hypertext. HTML is a markup language which is used to create static web pages. If you are thinking of creating your own web pages, you need to know at least basic HTML. These HTML documents are plain text files; user can create these documents using text editor like Notepad or gedit.

HTML is a hypertext Language because it supports font styled text, pictures, graphics and animations and also it provides hyper links that used to browse the Internet easily. Text becomes hypertext with the addition of links that connects other hypertext documents. Hypertext is a text augmented with links-pointers to other pieces of text, possible else where in the same document (internal linking) or in another document (external linking).

Note: Hypertext is non-linear text which contains links to other pieces of text Markup comprises a set of symbols that can be used to provide instructions for viewing a web page

History of HTML

HTML was originally developed by Tim Berners-Lee while at CERN, and popularized by the Mosaic browser developed at NCSA. HTML standards are created by group of interested organizations and individuals called **W3C (World Wide Web Consortium).** There have now been three official HTML standards:

1. HTML 2.0 was released in 1994 and remains the baseline for backwards compatibility and should be supported by all browsers.
2. HTML 3.2 was released in 1996 with many useful additions
3. HTML 4.0 was released in 1997 and slightly amended in 1999
4. Now, The current version is HTML 5.0 was released in 2012

1. First advantage it is widely used.

2. Easy to learn and use.

3. It is error-free language

4. Every browser supports HTML language.

5. It is by default in every ***OS.*** so you don't need to purchase extra software.

Disadvantages of HTML:

1. It can create only static and plain pages so if we need dynamic pages then HTML is not useful.

2. Need to write lot of code for making simple webpage.

3. Security features are not good in HTML.

2.2 Tags:

A tag is a format name surrounded by angular brackets which tells when to switch on a piece of formatting and when to switch it off.

- A tag is made up of left operator (<), a right operator (>) and a tag name between these two operators.
- If you forget to mention the right operator (>) or if you give any space between left operator and tag name browser will not consider it as tag.
- At the same time if browser not understands the tag name it just ignores it, browser won"t generate any errors.
- HTML language is not case sensitive; hence user can write the code in either upper case or lower case. No difference between <HTML>, <html> and <hTmL>
- Every HTML document begins with start tag is <HTML> terminates with an ending tag is </HTML>

- HTML documents should be saved with the extension **.html**. Syntax of a tag:tagname [parameter=value]>

Ex: HR is a tag name that displays a horizontal ruler line.
HR> - ---- (No parameters, no value)
<HR ALIGN=CENTER> ----- (Tag with parameter and value for the parameter)
<HR WIDTH=”30%” SIZE=”100” ALIGN=”RIGHT”> ----- (Tag with more parameters with their values)

Different types of Tags:

1.
1. Container/Paired tags

1.
2. Empty/Singleton tags

- Singleton tag does not require an ending tag but they end with slash. (Ex:<HR/>)
- Container tag required an ending tag, which is similar to opening tag except backslash before the tag name (Ex: <HTML> is opening tag, then ending tag is </HTML>

2.3 Structure of an HTML document:

All HTML documents follow the same basic structure. They have the root tag as <html>, which contains <*head*> tag and <*body*> tag. The <head> tag is used to place control information used by the browser and server. The <body> tag contains the actual user information that is to be displayed on the computer screen. The basic document is shown below:

<*html*>
<*head*>
<*title*> *Basic HTML document* </*title*>
</*head*>

<body>

*<h1> Welcome to the world of Web Technologies</h1> A Sample HTML Document *

</body>

</html>

Besides <head> and <body> tag, there are some other tags like title, which is a sub tag of head, that displays the information in the title bar of the browser. <h1> is used to display the line in its own format i.e., bold with some big font size. is used to write the content in the form of paragraph.

Comments in HTML

An HTML comment begins with "<!--" and ends with "-->". There should not be a space between angular bracket and exclamation mark. Each comment can contain as many lines of text as you like. If comment is having more lines, then each line must start and end with -- and must not contain -- within its body.

<! -- this is a single line comment line - ->

! -- this is a multiline comment ---- spawned over ---- three line -->

The major purpose of comments to improve the readability of a program.

The Document Head

The only tag that most authors insert in their head sections is the title:

<title>. . . </title>

All HTML documents have just one title which is displayed at the top of the browser window. The title is used as the name in the bookmark fields and on search engines

The Document Body

In HTML, Documents are structured as blocks of text, each of which can be formatted independently. The two major block of text in HTML documents are the paragraph and heading.

In HTML paragraphs are created by tag and headings are created by <hn> tag where n starts from 1 to 6

Creating HTML Page

The Following steps are needed to create a HTML page

Step 1: Open any text editor like Notepad, gedit, Wordpad etc.

Step 2: Use the File menu to create a new document (Fileà New) and type the HTMLcode

<HTML>

<HEAD>

<TITLE>First page </TITLE>

<HEAD>

<BODY>

Hi… This is my first Webpage - *Samrat.*

</BODY>

</HTML>

Step 3: Go to the file menu and choose*"Save as"*option (File->Save as) and give thename of the file as "example.html" under root directory(C:)(or any valid path)

Step 4: Double click to execute it. The output displayed as follows

Basic HTML tags

Character-formatting tags:

You can use character formatting tags to format a text block that is as small aa a single character or as large as an entire document.Some of the most frequently used Character-formatting tags are :

1. Boldface tag

This tag is used for implement bold effect on the text …….

2. Italic tag

This implements italic effects on the text.

…….

3. Underline tag

This is used to specify that the selected text be displayed with underline.

<u>. . . </u>

4. Big tag

This is to specify that the selected text be displayed as bigger font size as compared to the font for the rest of the text.

<big>. . .</big>

5. Small tag

This is to specify that the selected text be displayed as smaller font size as compared to the font for the rest of the text.

<small>. . .</small>

6. strong tag

This tag is used to always emphasized the text

. . .

7. sub and sup tag

These tags are used for subscript and superscript effects on the text.

_{. . .}

^{. . .}

Block-formatting tags:

You can use following tags to format blocks of text within HTML document. Some of the most frequently used Block-formatting tags are :

1. Body tag (<body>)

Body tag contains some attributes such as bgcolor, background etc. *bgcolor* is used for setting the background color of a webpage which takes background color name or hexadecimal number such as #000000 to #FFFFFF and background attribute used for setting mage as a background for webpage and it will take the path of the image which you can place as the background image in the browser.

Syntax:

<body bgcolor=”name/#rrggbb” background=”image name”> . . . </body>

Example:

<body bgcolor=”#F2F3F4” background= “c:\samrat\image.jpg”>...</boby>

2. Paragraph tag ( )

Most text is part of a paragraph of information. Each paragraph is aligned to the left, right or center of the page by using an attribute called as align. Syntax:

<p [align=”left” | “right” | “center”] > . . . **3. Heading tag(<Hn>)**

HTML is having six levels of heading that are commonly used. The largest heading tag is <h1>. The different levels of heading tag besides <h1> are <h2>, <h3>, <h4>, <h5> and <h6>. Each heading tag has an attribute called as align which can be set to left, center, or right. By default all headings align left.

Syntax:<h1 [align=”left” | “right” | “center”]> . . . </h1> <h2 [align=”left” | “right” | “center”]> . . . </h2>

<h6 [align=”left” | “right” | “center”]> . . . </h6>

4. <hr> tag

This tag places a horizontal line across the screen. These lines are used to break up the page. This tag does not require an end tag. This tag also contains attributes which determines how the rule will be displayed. It can be aligned but by default is centered on the screen. The *size* attribute specifies the thickness of the rule in pixels. *noshade* draws the rule as a single thick line rather than giving it‟s default 3Dappearance.

Syntax:

<hr align=”left” | “right” | “center” width=”nn%” size=”n” [noshade]/>.

5. base font

This specify the minimum text size for the entire webpage but not the headings.

Syntax:<basefont size=”n”>

font tag

This sets font type, size, color and relative values for a particular text. Absolute font sizes are can be set from 1 to 7. Relative font sizes are set by using +/- 1 to 7. The color of the text is set by *color* attribute. This takes hex value which represents the amounts of red, green and blue in a chosen color.

Syntax:

6. tt tag

This tag is used to give teletype effect on the text

<tt>. . .</tt>

7. Line break tag

This tag is used to the break the line and start from the next line. It is an empty tag.

**
**

8. Pre-formatted text tag

It Considers spaces, new lines etc. and as it is prints the information.

<PRE>. . .</PRE>

Example : HTML code to implement common tags.

mypage.html

<html>

<head> <! -- This page implements common html tags --> <title> My Home page </title>

</head>

<body >

<h1 align="center"> P.B.Siddhartha College of Arts & Science</h1> <h2 align="center"> Vijayawada</h2>

<basefont size=4>

<p> This college runs under the <tt>Siddhartha Adademy of General and Technical Education</tt> of &quot Vijayawada&quot &amp

it is affiliated to Krishna University

<hr size=5 width=80%>

<h3> <u>&lt Some common tags &gt</u> </h3>

</body>

</html>

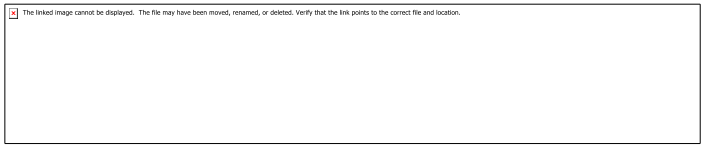


Fig 1.3.1: Example Output

9. MARQUEE Tag:

This is to create scrolling effect for the selected text in an HTML Page. When you use this tag, the selected text is highlighted and scrolls from right to left and vice versa. Syntax:

<marquee> . . .</marquee>

Marquee tag attributes:

a. align: sets the alignment of the text relative to the marquee. Set to top(default), middle or bottom.

b. behavior: Sets how the text in the marquee should move, It can be scroll(default), slide(text enters from one side and stops at the other end), or alternate(text seems to bounce from one side to other)

c. bgcolor: sets the background color for the marquee box

d. direction: sets the direction of the text for scrolling. It can be left(default), right, down or up.

e. height and width: to specify the height and width of a marquee text either in pixels or in % of the screen height.

f. Loop: to specify the number of times a marquee will loop. -1 for infinite times.

g. scrolldelay: to control the speed at which marquee text is drawn on the screen. This attribute specifies the number of milliseconds between each successive draw of the marquee text.

h. scrollamount: to specify the number of pixels that marquee text moves in every movement.

Example:

<marquee align="middle" behavior="slide" bgcolor="red" direction="down"> Samrat

</marquee>

10. Blinking text Tag

<blink>……</blink> displays enclosed text as blinking on and off approximately once a second. This tag only works in Netscape navigator

11. Escape sequences

These are character escape sequence which are required if you want to display characters that HTML uses as control sequences. All of these characters starts with an ampersand **(&)** and are terminated with a semicolon **(;)**

&< >   " © & < > ‘ ©

2.4 HYPERLINKS

The real power and flexibility of HTML is in Hyperlinks. Hyperlinks are created with anchor tag**(<a>)**

You can use hyperlink to

1. create links with HTML pages

2. link different HTML pages

3. access services at other Internet sites

Anchor tag:

The anchor tag is created by <a> . . . tags. The tag has three sections: the address of the referenced document, a piece of text displayed as link, and the closing tag. Syntax:

 Text

- *href* means Hypertext references that can be used for giving the path of a file whichyou want to link.

- The “Text” between the <a> . . . tags acted as a hyperlink. This text is called *“hypertext”.* When you click on this text, the linked page or file will be displayed. You can also replace this text with image. In that case, the image will act as a hyperlink.

Attributes

href

Used to specify the path and file name of the HTML page that you need to access by using a hyperlink

The *name* attribute enable you to create anchor with in HTML page. This anchor tag is used to bookmark a location in an HTML page.

The *title* attribute is used to provide information about a link. The value ***title*** specified for this attribute appears as a small tip when you place the mouse pointer over the hyperlink.

The *target* attribute is used to specify the target window where the linked ***Target*** HTML page is displayed. You can assign the name of a frame as the value for this attribute.

Creating hyperlinks to HTML Pages

To create hyperlinks to an HTML page, you specify the address of the HTML pages as the value of *href* attribute of the <a>. . . tags

Consider the example of simple HTML page that displays „next“ as hyperlink, Create a HTML page that connect web page created through hyperlink.

first.html

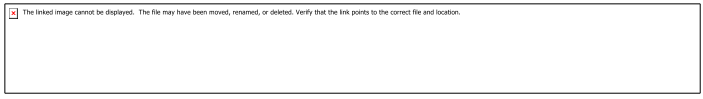


Fig 1.4.1 First.html

second.html

<html>

<head> <title> Navigation </title> </head>

<body>

You are in second pagePrevious Page

</body>

</html>

When you click on Next page, it opens

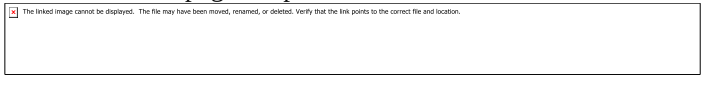


Fig 1.4.2 Second.html

Creating Hyperlinks within an HTML Page

HRFF and NAME are the most important attributes of anchor tag that are required to create hyperlinks within an HTML page. The *name* attribute is used to create anchors with in a page and the href attribute is used to create links that to these anchors.

To Create anchors in an HTML page, you can use the following syntax:

 This is anchor

Afte you create an in HTML page, you create a hyperlink to refer that anchor, To do this , use the following syntax:

 This is anchor

The hash (#) symbol before Myname specifies that it is the name of an anchor. When you click the text within the <a>. . . tags, The browser searches for the Myname anchor within the HTML page. When the anchor is found, the Webpage is scrolled up or down to display the text at the anchor position.

Consider the following code:

<html>

<head> <title> Anchors </title> </head>

<body>

* Top of the page Bottom *

*

*

* Bottom of the page Back to Top *

</body>

</html>

Output:

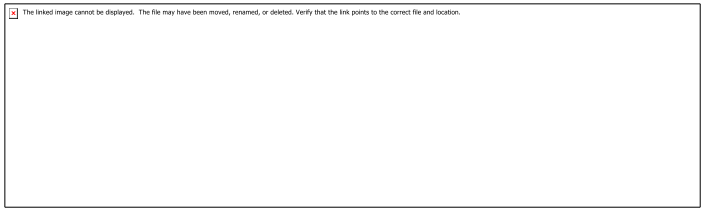


Fig 1.4.3 Document Linking

2.5 Color and Images

Color is essential to the Web experience; it brings life to pages and makes them exciting. Color can be used in number of places on a Webpage: The background can be colored, individual elements can be altered, and links which already colored can be changed.

To change the colors of links or of the page background hexadecimal values are placed in the <body> tag:

Syntax:

<body bgcolor = “#rrggbb” text = “#rrggbb” link= “#rrggbb” vlink= “#rrggbb” alink = “#rrggbb”>

The *vlink* attribute sets the color of links visited recently, *alink* the color of a currently active link. The six figure hexadecimal values must be enclosed in double quotes and preceded by a hash(#).

Images

Images are second aspect of pleasant Web experience. You can add images to an HTML page to either improve its appearance or present important information. To

add images to an HTML page, you can use several image formats. These formats include *„gif", „jpg" and „png".*

Images can be added in two different ways:

1. By using "*background"* attribute of <body> tag

2. By using <img/> tag

The <BODY> Tag: background attribute

You can use the background attribute of the <BODY> tag to add image as a background in an HTML page.

Syntax:

<body background = "URL">. . .</body>

This tag will set a background image present in the URL.

The Image tag

You can use image tag in an HTML page to add images along with text. An image added using the image tag occupies space within the HTML page. To add image to HTML page, use <*img*> tag.

Syntax:

<img src="URL" height="n" width="n" align = "left" | "right" | "top"|"middle" alt = " string ">

Attributes:

a. src : used to specify the name of the file. When using <img> tag, it is mandatory to specify a value for the **SRC** attribute

b. align : used to specify the vertical alignment of an image

c. height : used to specify the vertical area that an image will occupy in HTML page

d. Width : used to specify the horizontal area that an image will occupy in HTML page

e. alt : used to specify the text when browser unable to display the image or image not available.

Example : HTML code that implements color and image

```
<html>
<head>
<title> Colors and images </title>
</head>
<body bgcolor="gray" text = "silver" link = "white" vlink = "yellow" alink = "brown"> <a href="first.html">Go to First Page</a>
<img src= "C:\Users\Public\Pictures\Sample Pictures \Tulips.jpg" align ="middle" height = "100" width="100" alt="Tulips"/>
</body> </html>
```

Output:

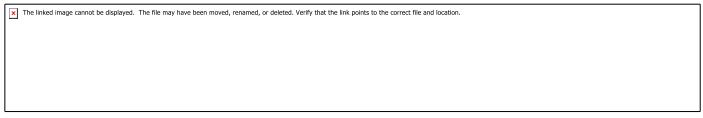


Fig 1.5.1 implements color and image

2.6 Lists

One of the most effective ways of structuring a web site is to use lists. Lists provides straight forward index in the website. HTML provides three types of lists.

1. Unordered / Bulleted list

2. Ordered / Numbered list and

3. Definition/Description list

Lists can be easily embedded within other lists to provide complex yet readable structures.

**
. . .:**

The Ordered / Unordered lists are each made up of set of list items. These list items are added with ***<li>*** tag. Elements of a list item may format with any of the usual formatting tags and may be images or hyperlinks.

Unordered Lists

Unordered lists are also called Un-numbered lists. The Unordered list elements are used to represent a list of items, which are typically separated by white space and/or marked by bullets. Using <UL> tag is used to create unordered lists in HTML. Which is paired tag, so it requires ending tag that is </UL>. The list of items is included in between <UL>…..</UL>. The *TYPE* attribute can also be added to the <UL> tag that indicates the displayed bullet along with list of item is **square, disc** or **circle**. By default it is disc.

Syntax:-

<UL [TYPE=”square | disc |circle “] >
<LI>item name1 </LI>
<LI>item name2 </LI>

<LI>item name n </LI>
</UL>

Example: Write a HTML program for displaying names of B.Tech Courses with defaultbullets and names of PG Courses with square bullets.

```
<html>
<head>
<title>Unordered Lists</title>
</head>
<body>
<h3>B.Tech Courses </h3>
<ul type="disc">
<li>CSE </li>
<li>IT </li>
<li>ECE</li>
<li>EEE</li>
<li>MECH</li>
```

*
*
<h3>PG Courses </h3>
<ul type="square">
*
M.Tech
*
*
MCA
*
*
MBA
*
*
*
</body>
</html>

Output:

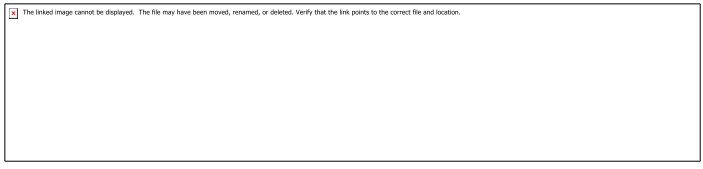


Fig 1.6.1 Unordered List

Ordered Lists:-

Ordered lists are also called **Numbered** or **Sequenced lists**. In the ordered list the list of items have an order that is signified by numbers, hence it some times called as number lists. Elements used to present a list of items, which are typically separated by white space and/or marked by numbers or alphabets. An orders list should start with the <OL> element, which is immediately followed by a <LI> element which is same as <LI> in unordered list. End of ordered lists is specified with ending tag </OL>.

Different Ordered list types like roman numeral list, alphabet list etc. can be specified with *TYPE* attribute. Another optional parameter with <OL> tag is *START* attribute, which indicates the starting number or alphabet of the ordered list. For example TYPE=”A” and START=5 will give list start with letter E. The TYPE attribute used in <LI>, changes the list type for particular item. To give more flexibility to list, we can use VALUE parameter with <LI>tag that helps us to change the count for the list item and subsequence items.

Syntax:-

<OL [TYPE=“1 |A |a | I|i] start=”n”>

<LI>item name1
<LI>item name2

<LI>item name n
</OL>
Different Ordered list types
Type=”1” (default) e.g.1,2,3,4…..
Type=”A” Capital letters e.g.A,B,C…
Type=”a” Small letters e.g. a,b,c……
Type=”I” Large roman letters e.g. I, II, III,…

Example: Write a HTML program for displaying Colors of VIBGYOR with numbers and names of Fruits with alphabets from “E”.

```
<html>
<head> <title>Ordered Lists</title> </head>
<body text="maroon">
<h3>Colors in VIBGYOR</h3>
<OL>
<LI>Violet</LI>
<LI>Indigo</LI>
<LI>Blue</LI>
<LI>Green</LI>
<LI>Yellow</LI>
<LI>Orange</LI>
<LI>Red</LI>
</OL>
<h3>Types of Fruits</h3>
<OL type="A" start="5">
<LI>Apple</LI>
<LI>Banana</LI>
<LI>Grapes</LI>
</OL>
```

</body>
</html>

Output:

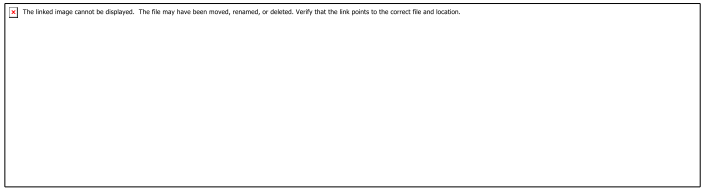


Fig: 1.6.2 Ordered Lists

Nested Lists: Lists can be nested that is Nested Lists is list with in another list.

2.7 Tables

Table is one of the most useful HTML constructs. Tables are finding all over the web applications. The main uses of table are that they are used to structure the pieces of information and to structure the whole web page.

- Tables allow you to present data across rows and columns, making it easy for reader to correlate connected pieces of information.
- A Table is collection of rows and columns or collection of cells
- The basic element in a table is *cell* or the *grid.* A cell is formed when a row crosses a column.
- You can specify the amount of space between cells in a table. This space is called *cell spacing*
- You can specify the amount of space between the contents of a cell and the cell wall. This space is called *cell padding*

You can use various elements to specify the details of a table. Many table elements also take attributes, which allows you to further specify the look of the table.

- TABLE
- Table row
- Table data
- Table Heading

The TABLE Element

The TABLE element is the container element for table and uses the <TABLE>...</TABLE> tags to enclose all the other table tags. If the <TABLE> tag is omitted or not closed, the browser ignores all the other tags that you specify for the table. Everything that we write between these two tags will be within a table. The attributes of the table will control in formatting of the table.

The <TABLE> tag has the following attributes:

a. align: used to specify the alignment of a table in a HTML page.

b. border: used to specify the thickness of the table border in pixels.

c. bgcolor: used to specify the background color for the table.

d. frame: used to specify the which side of the outer border is visible in the browser. You must specify the border attribute before you specify the FRAME attribute. *Values for the FRAME Attribute*

- VOID-Removes all External borders
- BOX-Displays a box around the table i.e. all four sides of the table
- HSIDES-Displays an external border at the top and bottom of the table
- VSIDES-Displays an external border at the right and left of the table
- LHS-Displays an external border only on the left hand side of the table
- RHS-Displays an external border only on the right hand side of the table
- ABOVE-Displays an external border only on the top of the table
- BELOW-Displays an external border only on the bottom of the table
- cellspacing: used to specify the spacing between cells in pixels.

g. cellpadding: used to specify the spacing between cell content and cell wall in pixels.

h. height: used to specify the height of table in pixels or %.

i. width: used to specify the width of table in pixels or %.

Syntax:

<TABLE align="left | right | center" border="n" bgcolor="#rrggbb" cellspacing="n" cellpadding="n" frame="value" rules="value" height="n | %" width="n | %">

. . .

</TABLE>

The Table row element

You use table row element to create rows in a table. The Table row element uses the <TR> tag to create a row. You can use <TR> tag within <TABLE> tags. <TR> tag used as container for the row.

The <TR> tag has the following attributes:

a. align: used to specify the horizontal alignment of the contents for cells of a row.

b. valign: used to specify the vertical alignment of the cell content for all cells of the row.

c. bgcolor: used to specify the background color of the row.

Syntax:

<TR *align="left | right | center" align="top | bottom |middle" bgcolor="#rrggbb"*> ….. </TR>

The Table data element

You use the Table data element to create data cells. The Table data element uses <TD> tag within the <TABLE> and <TR> tags to create data cells. A data cell only appears with in a table row.

The <TD> tag has the following ***attributes:***

a. colspan: used to specify the number of columns the cell can span.

b. rowspan: used to specify the number of rows the cell can span

c. align: used to specify the horizontal alignment of the data within a cell.

d. valign: used to specify the vertical alignment of data within the cell.

e. bgcolor: used to specify the background color of the cell.

Syntax:

<TD *align=”left | right | center” align=”top | bottom |middle” bgcolor=”#rrggbb” colspan=”n” rowspan=”n”*>…..</TD>

The Table heading element

You use the table heading element to create heading cells for table rows, columns, or both. The table heading element uses <TH> tag to create heading cells. The <TH> tag is exactly same as <TD> tag, except that the text displayed in bold. <TH> tag and <TD> tag share the same properties.

Example: HTML Program that prints table

```
<html>
<head> <title> table page</title> </head>
<body>
<table align="left" cellpadding="2" cellspacing="2" border="3" bgcolor="red">
<tr>
<th colspan="4"> Sample Table</th>
</tr>
<tr>
<th>HEAD1</th>
<td rowspan="3">CELL2</td>
<th>HEAD3</th>
<th>HEAD4</th>
<tr>
<td>CELL1</td>
<td> CELL3</td>
<td> CELL4</td>
```

```
</tr>
<tr>
<td>CELL5</td>
<td colspan="2" align="center"> CELL6</td>
</tr>
</table>
</body> </html>
```

Output:

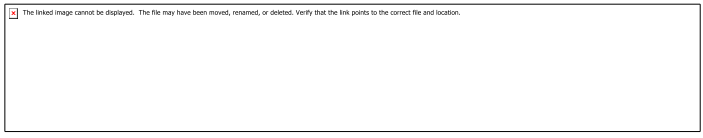


Fig 1.7.1 Table

2.8 Frames:

Frames provide a pleasing interface which makes your web site easy to navigate. You can use frames in HTML page to manage the layout of the page. Frames divide the browser window into horizontal or vertical sections. Each section can be used to display a separate HTML page.

Each frame within a webpage has the following properties:

- It has unique name
- It displays an HTML document independent of other frames
- Its size can be dynamically changed according to the size of the content in HTML page

Creating frames

When we talk about frames actually we are referring to frameset, which is a special type of web page. Simply frameset is nothing but collection of frames. Web page that contains frame element is called framed page. Framed page begins with <frameset> tag and ends with </frameset>. Each individual frame is identified through <frame> tag. Creation of framed page is very simple. You can nest the framesets. First you decide how you want to divide your webpage and accordingly define frame elements.

Consider the following diagrams, first page divides into two columns and the second form divides into three rows.***Two columns framesetThree rows frameset***

The <FRAMESET> . . . </FRAMESET> Tags

The <FRAMESET> . . . </FRAMESET> tags are the container tags for all other tags that are used to create frames. These tags are used to specify the size and number of the frames to be created in an HTML Page. The frames within these tags are jointly referred to as a *frameset.*

Syntax:

<FRAMESET cols=”n,n” rows=”n,n”>

<FRAME> Tags

</FRAMESET>

Attributes

COLS: The COLS attribute is used to specify the number of vertical frames within HTMLPage.

ROWS: The ROWS attribute is used to specify the number of horizontal frames withinHTML Page. To specify the values of the attributes, use a comma-separated list of values. The values can be of the following two types:

- Value: This numeric value is used to specify number of pixels that a frame will occupy in the browser window.

- Value%: This value is used to specify percentage of space that each frame will occupy in the browser window. The total of all the values specified in the comma-separated list is 100%.

In order to divide into two columns we can use the following syntax *<FRAMESET COLS=”25%,75%>*

<frame> tags

</FRAMESET>

In the second diagram we have three rows so by using rows parameter of frameset, we can divide logically the window into three rows.

<FRAMESET ROWS=”20%,70%,10%>

<frame> tags

</FRAMESET>

According to above code, first row occupies 20% of the window, third row occupies 10% of the window, second row occupies 70% of the window.

<FRAME> Tag

You use the <FRAME> Tag to create frame with in *frameset*. You can use the following attributes to specify the name of the frame and the HTML page that you need to display in the frame.

1. *SRC*: SRC attribute to specify the HTML page that you want to display in a frame.
2. *NAME*: NAME attribute to specify the name of the frame. This *NAME* is used bythe anchor element to display the linked HTML Pages within frame.
3. *SCROLLING*: attribute used to add scrollbars to a frame. The SCROLLING attribute takes three value: YES,NO, AUTO.

- The value *YES* specifies that a scrollbar should always be visible on the frame
- The value *NO* specifies that the scrollbar should never be visible on the frame
- The value *AUTO* specifies the browser to automatically display or remove scrollbars from a frame

1. *FRAMEBORDER:* attribute to specify whether a frame should have a border. Thevalue 0(zero) specifies no border. If you specify any other numeric value, the border is displayed.
2. *NORESIZE:* By default, You can resize a frame by dragging its borders. You canrestrict this by specifying the NORESIZE attribute.

Syntax:

<FRAME SRC = ”URL” NAME =” myframe” SCROLLING = ”yes | no | auto” FRAMEBORDER = ”0|1” [NORESIZE]/>

2.9 Forms

Forms are the best way of adding interactivity to a web page. A form is a collection of fields that you can use to gather information from people visiting your site. Forms acts as a means of user interactions on the Web.

Working of a Form

A Form contains certain text fields, radio buttons, check boxes and buttons for entering data. When data entry is complete, the user submits the form for processing using SUBMIT button. This communicates to the browser that the user has completed data entry. The browser sends data to the server. The server process the details and sends response to browser, which displayed on the browser.

The FORM Tag

A form is defined using the <FORM>. . . </FORM> tags. The FORM tag has four attributes:

1. NAME
2. ACTION
3. METHOD
4. ENCTYPE Syntax:

<FORM action=”URL” method = “POST” | “GET” enctype=”encoding”> . . . </FORM>

NAME: used to specify the name of the form

ACTION: The ACTION attribute of the form tag is used to specify the name, and locationof a script that will be used to process the data.

METHOD: The METHOD attribute of the form tag is used to specify the method by whichbrowser sends the data to the server for processing. The type method can be either GET or POST. When *get* is used, the data is included as part of the URL. The *post* method encodes the data within the body of the message. Post can be used to send large amount of data, and it is more secure than *get*.

ENCTYPE: The ENCTYPE attribute is used to specify the content type for encoding thedata. There are two content types:

1. application/x-www-form-urlencoded(default)

2. multipart/form-data.

ENCTYPE is used only when you need to upload a file.

The INPUT Element

The INPUT Element is used to specify input fields, such as single line text fields, passwords fields, check boxes, radio buttons etc.,. The INPUT element is defined by using <INPUT> tag.

You can create the following controls using the INPUT tag:

- Text
- Password
- Radio
- Checkbox
- Button
- Submit
- Reset

Syntax:

<INPUT type = “text” | “password” | “checkbox” | “radio” | “submit” name=”string” value=”string” size=”n” maxlength=”n” />

The tags used inside the form tag are:

In the above tag, the attribute *type* is used to implement text, password, checkbox, radio, submit, reset button and button.

Text: It is used to input the characters of the size n and if the value is given than it is used as a default value. It uses single line of text. Each component can be given a separate name using the name attribute.

Password: It works exactly as text, but the content is not displayed to the screen, instead * is displayed.

Radio: This creates a radio button that accepts single value from a set of alternatives.

They are always grouped together with a same name but different values.

Checkbox: This creates a checkbox that enables you select multiple values from a set of alternatives. They are always grouped together with a same name but different values.

Submit: This creates a Submit button which displays the value attribute as its text. It is used to send the data to the server.

Reset: This creates a Reset button which displays the value attribute as its text. It is used to clear the form data.

Button: This creates a user-defined button which displays the value attribute as its text.

It works according to user requirement.

The SELECT Element

The SELECT Element is a container element. This element is used to create a list box or drop-down list box.

Syntax:

<SELECT NAME=”string” SIZE=”n” MULTIPLE> . . .</SELECT>

Where

MULTIPLE: Indicates that the user is allowed to make several selections from list box

*NAME: repre*sents name of the list box

SIZE: Specifies the number of visible items in a list box.

The OPTION Element

It represents a choice in the list box and only occurs in SELECT element.

Syntax:

<OPTION VALUE=”String” SELECTED>……</OPTION>

SELECTED: indicates the option is initially selelcted

VALUE: value that indicates a particular option selected.

Example:

```
<html>
<body>
<form>
<select name="flavour">
<option value="1" selected>Vanilla</option>
<option value="2">Strawberry</option>
```

<option value="3">Chocolate</option>
<option value="4">Peach</option>
</select>
</body>
</html>

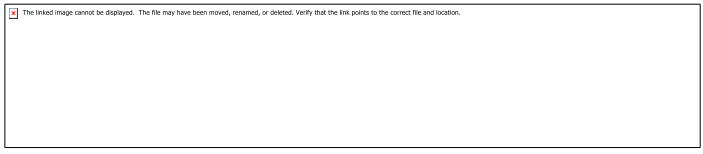


Fig 1.9.1 Option Element

TEXTAREA Element

This creates a multi-line plain text field into which the user can enter anything they like.

The area will be sized at rows by cols but supports automatic scrolling.

Syntax:

<TEXTAREA NAME=”string” ROWS=”n” COLS=”n”> . . . </TEXTAREA>

Where

NAME: Specify the name of the text field

ROWS: Set the number of rows of text that will be visible

COLS: Set the number of columns of text that will be visible

Example : HTML code that implements forms

<html>

<head> <title>Student Data </title> </head>

<body>

<form>

*<p><td>Name:<input type="text" maxlength="30" size=15>
 Password:<input type="password" maxlenght=10 size=15>
 Qualification:<input type="checkbox" name="q" value="bt">B.Tech <input type="checkbox" name="q" value="mt">M.Tech
 Gender:<input type="radio" name="g" value="m">Male*

*<input type="radio" name="g" value="f">Female
 Course:<select name="course" size="1">*

<option value="1" selected>CSE

<option value="2">ECE

*</select>
*

*Address:<textarea name="addr" rows="5" cols="15"></textarea>
*

<input type="submit" name="s" value="Accept">

<input type="reset" name="c" value="Ignore"></p> </form>

</body>

</html>

Output:

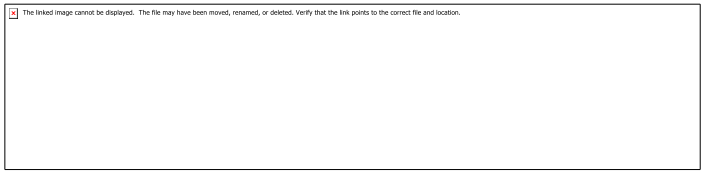


Fig 1.9.2 Forms

2.10 Cascading Style Sheets

CSS stands for Cascading Style Sheets. A style sheet is document that allows youto control the rendering, such as fonts, colors, typefaces and other aspects of style, of a web document. You can define style sheet properties within HTML document or in an external file.

- A Style sheet is a collection of rules

Advantages:

1. Enables you to separate content from formatting.
2. Reduces download time by removing formatting information from the document.
3. Allows you to ensure a consistent appearance across a site
4. It supports reusability (Inheritance).

Defining a style:

A rule is a statement that defines a style.

Property name

Property value
{
color:red;
}Selector
Declaration
A rule consists of two parts:

1. **Selector**

2. **Declaration**

A selector is an HTML statement that is linked to a specific style. A declaration defines the style for selector. A declaration again has two parts:

1. **Property name**

2. **Property value**

Property name is name of the property for which you need to define a style. Property value is the value assigned to property name.

Types of style sheets

You can define three types of style sheets:

1. Inline style sheets
2. Internal/Embedded style sheets
3. External style sheets

An inline sheet applies style to a particular element in a webpage. Embedding a style sheet is for defining styles to a single webpage. An external style sheet can apply styles to multiple web pages.

Inline style sheets

The style included within the tag is known as *inline style sheet*. To include style definition within tag, you use "STYLE" property.

Syntax:

<TAG STYLE=”propertyname: property value;”>. . . </TAG>

An Inline style affects only the element for which the style is defined. The style does not affect any other element even if the element is of the same type.

Consider the following code:

<html>

<head>

<title> Inline style sheets </title>

</head>

<body>

<P STYLE=”font-family: Arial; font-style: italic; color: red;”>This is paragraph with inline style </P>

<P>This is paragraph without style </P>

</body>

</html>

Output

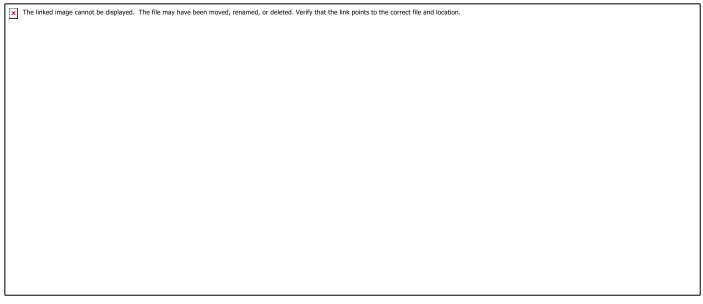


Fig 1.10.1 Inline Style Sheet

Internal style sheets

In this, the style definitions are enclosed in <STYLE> . . . </STYLE> tags. In turn, this STYLE tag pairs must be placed in between <HEAD> . . . </HEAD> tags. The style defined in <HEAD> section is applied to whole document. The styles defined for the tags within <STYLE> tag will reflect for every instance of the tag in the entire document.

Syntax:

<HTML>

<HEAD>
<STYLE TYPE=”TEXT/CSS”>
<!-- STYLE DEFINITIONS -->
</STYLE>
</HEAD>
<BODY> . . . </BODY>
</HTML>
onsider the following code that illustrates internal style sheets:

```
<html>
<head>
<title> Internal style sheets </title>
<STYLE>
h1 {
font-style:italic;
color:red;
background:yellow;
}
</STYLE>
</head>
<body>
<h1>An Embedded style sheet</h1>
<p> Embedded style sheet is a part of HTML document</p> </body>
</html>
```

Ouput:

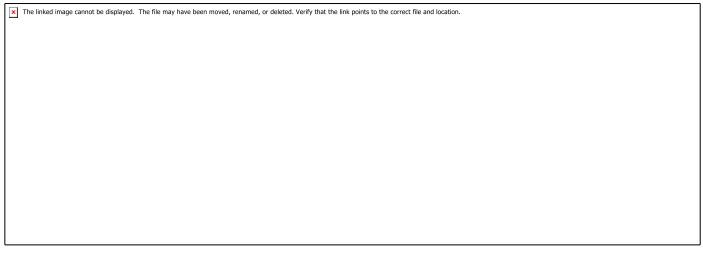


Fig 1.10.2 Embedded Style Sheet

External Style Sheets

Styles defined in the external files are called external style sheets which can be used in any document by including them via URL. Use an external style sheet when you want to apply one style to many pages. If you made any change in an external style sheet, the change is universal on all the pages where the style sheet is used. An external style sheet is declared in an external file with a *.css* extension. It is called by pages whose interface it will affect.

Advantages of External CSS

1. The same style can be used by all of the WebPages in your website.

2. You can change the appearance of several pages by just altering the style sheet rather each individual page.

3. A Style sheet can acts as style template

4. A style sheet can import and use styles from other style sheets, making modular development and good reuse.

5. Reduces the download time by removing formatting information from the document.

<LINK> Tag

External style sheets are called using the <*link*> tag which should be placed in the head section of an HTML document. This tag takes three attributes.

Attributes of the <link> tag:

- **rel** - When using an external stylesheet on a webpage, this attribute takes thevalue "stylesheet"
- **type** - When using an external stylesheet on a webpage, this attribute takesthe value "text/css"
- **href** - Denotes the name and location of the external style sheet to be used.

Syntax:

<link rel ="stylesheet" type="text/css" href="filename.css"> Consider the following code that illustrates external style sheets: ***styles.css***

```
h1
{
font-style:italic;
color:red;
background:yellow;
}
P
{
color:blue;
background:yellow;
font-size:12pt;
border:1px solid red;
}
```

Test.html

```
<html>
<head>
<title> External style sheets </title>
<link rel ="stylesheet" type="text/css" href="styles.css"> </head>
<body>
<h1>An External style sheet</h1>
<p> External style sheet is a master style sheet stored <br/>in an external file with .css extesion</p>
</body>
</html>
```

Output

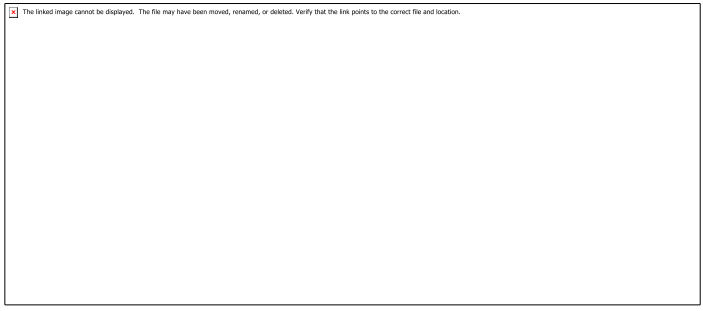


Fig 1.10.3 External Style Sheet

Class selectors

Class selectors are used, when you want different elements to share the same format. A class selector definition starts with a period **(.)** followed by user defined and

then style definition.

Syntax:

<style>

.myname

{

Style definitions

}

</style>

CHAPTER THREE

JavaScript

3.1. Introduction JavaScript

Script means small piece of code. Scripting languages are two kinds one isclient-side other one is servers-side scripting. In general client-side scripting is used for verifying simple validation at client side, server-side scripting is used for database verifications. VBScript, java script and J script are examples for client-side scripting and ASP, JSP, Servlets,PHP etc. are examples of server-side scripting.

JavaScript (originally known as "LiveScript") is a scripting language that runsinside the browser to manipulate and enhance the contents of Web pages. Java Script is designed to add interactivity to HTML pages. Web pages are two types

1. 1.Static web page-Static web page where there is no specific interaction with the client
2. 2.Dynamic webpage-Dynamic web page which is having interactions with client and as well as validations can be added.

Simple HTML script is called static web page, if you add script to HTML page it is called dynamic page. Netscape navigator developed java script. Microsoft's version of JavaScript is Jscript.

- Java script code as written between <script>-----</script>tags
- All java script statements end with a semicolon
- Java script ignores white space
- Java script is case sensitive language
- Script program can save as either. Js or. html

Benefits of JavaScript

- It is widely supported by web browsers;
- It gives easy access to the document objects and can manipulate most of them.
- Java Script gives interesting animations with long download times associated with many multimedia data types;
- Web surfers don't need a special plug-in to use your scripts
- Java Script relatively secure - you can't get a virus infection directly from Java Script.
- JavaScript code resembles the code of C Language; the syntax of both the language is very close to each other. The set of tokens and constructs are same in both the language.

Problems with JavaScript

- Most scripts relay upon manipulating the elements of DOM;
- Your script does not work then your page is useless

- Because of the problems of broken scripts many web surfers disable java script support in their browsers
- Script can run slowly and complex scripts can take long time to start up

Similarities between java script and java:

- Both java script and java having same kind of operators
- Java script uses similar control structures of java
- Nowadays both are used as languages for use on internet.
- Labeled break and labeled continue both are similar
- Both are case-sensitive languages

Operators in JavaScript:

- Arithmetic operators (+,-,*,/,%)
- Relational operators (<,>,!=,<=,>=)
- Logical operators(&&,||,!)
- Assignment operator(=)
- Increment decrement operators(++,- -)
- Conditional/Ternary operator(?:)
- Bitwise operators(&,|,!)

Control structures:

- If statement
- Switch
- While
- Do-while
- For
- Break
- Continue

Control structures syntax and working as same as java language.

3.2 Variables

Variables are like storage units/place holders to hold values. A variable is a memory location to hold certain different types of data. In Javascript, A variable can store all kinds of data. It is important to know the proper syntax to which variables must conform:

- They must start with a letter or underscore ("_")
- Subsequent characters can also be digits (0-9) or letters (A-Z and/or a-z).

Remember, JavaScript is case-sensitive. (That means that MyVar and myVar are two different names to JavaScript, because they have different capitalization.)

- You cannot use reserved words as variable names.
- You cannot use spaces in names.
- Names are case-sensitive.

Syntax:

var v_name = value;

Examples of legal variable names are fname, temp99, and _name.

When you declare a variable by assignment outside of a function, it is called global variable, because it is available everywhere in the document. When you declare a variable within a function, it is called local variable, because it is available only within the function. To assign a value to a variable, you use the following notation:

```
var num = 8;
var real= 4.5;
var myString = "Web Technologies";
```

Values of Variables(Data types)

JavaScript recognizes the following types of values:

- Numbers, such as 42 or 3.14159
- Boolean values, either true or false
- Strings, such as "Howdy!"
- NULL, a special keyword which refers to nothing.

3.3 Functions

Functions are one of the fundamental building blocks in JavaScript. A function is a JavaScript procedure -a set of statements that performs a specific task when called. A function definition has these basic parts:

- The *function* keyword
- A function name
- A comma-separated list of arguments to the function in parentheses
- The statements in the function in curly braces: { }

Defining a Function

Defining the function means, name the function and specifies what to do when the function is called. You define a function within the <SCRIPT>...</SCRIPT> tags within the <HEAD> ... </HEAD> tags. While defining a function, you can also declare the variables which you will be calling in that function. Here's an example of *defining* a function:

```
function msg()
{
window.alert("This is an alert box.");
}
```

Here's an example of a function that takes a parameter:

```
function welcome(string)
{
window.alert("Hi"+string);
}
```

When you call this function, you need to pass a parameter (such as the word that the user clicked on) into the function.

Calling a Function

Calling the function actually performs the specified actions. When you call a function, this is usually within the BODY of the HTML page, and you usually pass a parameter into the function on which the function will act.

Here's an example of calling the same function:

```
msg();
```

For the other example, this is how you may call it:

<inupt type="button" name=”welcome” onClick="*msg1(“Vijay”)*"/> **1.14 OBJECTS IN JAVA SCRIPT**

When you load a document in your Web browser, it creates a number of JavaScript objects with properties and capabilities based on the HTML in the document and other information. These objects exist in a hierarchy that reflects the structure of the HTML page itself. The pre-defined objects that are most commonly used are the window and document objects. Some of the useful Objects are:

1. Document
2. Window
3. Browser
4. Form
5. Math
6. Date

3.4 The Document Object

A document is a web page that is being either displayed or created. The document has a number of properties that can be accessed by JavaScript programs and used to manipulate the content of the page.

write or writeln

Html pages can be created on the fly using JavaScript. This is done by using the write or writeln methods of the document object.

Syntax:

document.write (“String”);

document.writeln (“String”);

In this document is object name and write () or writeln () are methods. Symbol period is used as connector between object name and method name. The difference between these two methods is carriage form feed character that is new line character automatically added into the document.

Exmaple: document.write(“<body>”);

document.write(“<h1> Hello </h1>”);

bgcolor and fgcolor

These are used to set background and foreground(text) color to webpage. The methods accept either hexadecimal values or common names for colors.

Syntax:

document.bgcolor=”#1f9de1”;

document.fgcolor=“silver”;

anchors

The anchors property is an array of anchor names in the order in which they appear in the HTML Document. Anchors can be accessed like this:

Syntax:

document.anchors[0];

:

document.anchors[n-1];

Links

Another array holding all links in the order in which they were appeared on the Webpage

Forms

Another array, this one contains all of the HTML forms. By combining this array with the individual form objects each form item can be accessed.

3.5 The Window Object

The window object is used to create a new window and to control the properties of window.

Methods:

1. *open(“URL”,”name”) :* This method opens a new window which contains thedocument specified by URL and the new window is identified by its name.

2. *close():* this shutdowns the current window.

Properties:

toolbar = [1|0] location= [1|0] menubar = [1|0] scrollbars = [1|0] status = [1|0] resizable = [1|0]

where as 1 means *on* and 0 means *off*

height=pixels, width=pixels : These properties can be used to set the window size.The following code shows how to open a new window

newWin = open("first.html","newWin","status=0,toolbar=0,width=100,height=100");

Window object supports three types of message boxes.

1. Alert box
2. Confirm box
3. Prompt box

Alert box is used to display warning/error messages to user. It displays a text stringwith *OK* button.

Syntax: window. Alert ("Message");

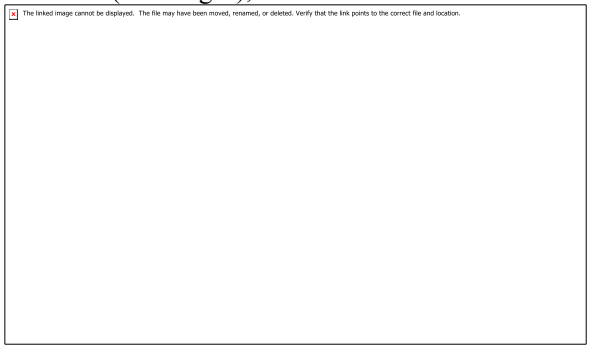


Fig: 3.4.1

Confirm Box is useful when submitting form data. This displays a window containing message with twobuttons: *OK* and *Cancel.* Selecting *Cancel* will abort the any pending action, while *OK* will let the action proceed.

Syntax

window.confirm("String");

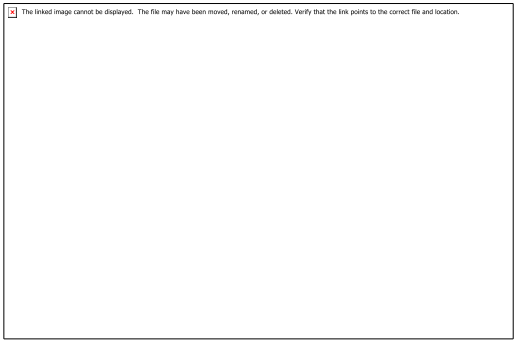


Fig: 3.4.2

Prompt box used for accepting data from user through keyboard. This displays simplewindow that contains a prompt and a text field in which user can enter data. This method has two parameters: a text string to be used as a prompt and a string to use as the default value. If you don‟t want to display a default then simply use an empty string. Syntax

Variable=window.prompt(“string”,”default value”);

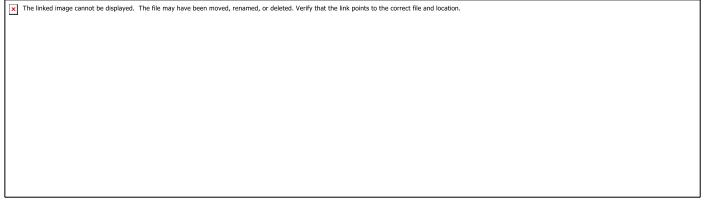


Fig:3.4.3

3.5 The Form Object

Two aspects of the form can be manipulated through JavaScript. First, most commonly and probably most usefully, the data that is entered onto your form can be checked at submission. Second you can actually build forms through JavaScript.

Form object supports three events to validate the form *onClick = “method()”*

This can be applied to all form elements. This event is triggered when the user clicks on the element.

onSubmit = “method()”

This event can only be triggered by form itself and occurs when a form is submitted.

onReset = "method()"

This event can only be triggered by form itself and occurs when a form is reset.

Example: HTML program that applies a random background color when you click on button

```
<html>
<head>
<script language = "javascript">
function change()
{
var clr = document.bgColor=parseInt(Math.random()*999999);
document.f1.color.value=clr;
}
</script>
</head>
<body>
<form name="f1">
<input type="text" name="color"/>
<input type="button" value="Cick me" onclick="change()"/> </form>
</body></html>
```

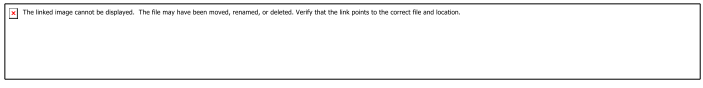


Fig: 3.5.1 On first run background is white

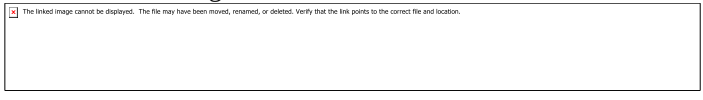


Fig: 3.5.1 After clicking onbutton"Click me"

3.6 The Math Object

The *Math* object holds all mathematical functions and values. All the functions and attributes used in complex mathematics must be accessed via this object.

Syntax:

```
Math.methodname();
Math.value;
```

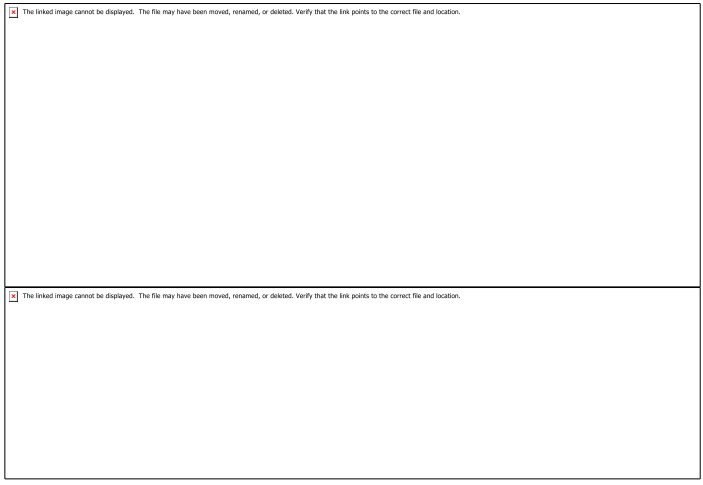


Fig: 3.6.2 The Math Object

3.7 The Date Object

This object is used for obtaining the date and time. In JavaScript, dates and times represent in milliseconds since 1st January 1970 UTC. JavaScript supports two time zones: UTC and local. UTC is Universal Time, also known as Greenwich Mean Time(GMT), which is standard time throughout the world. Local time is the time on your System. A JavaScript *Date* represents date from -1,000,000,000 to -1,000,000,000 days relative to 01/01/1970.

Date Object Constructors:

new Date(); Constructs an empty date object.

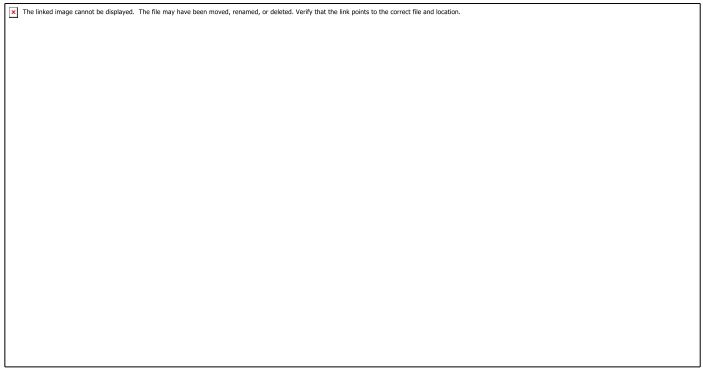
new Date("String"); Creates a Date object based upon the contents of a text string.

new Date(year, month, day[,hour, minute, second]); Creates a Date object based upon the numerical values for the year, month and day.

```
var dt=new Date();
document.write(dt);
// Tue Dec 23 11:23:45 UTC+0530 2015
```

Methods in Date object:

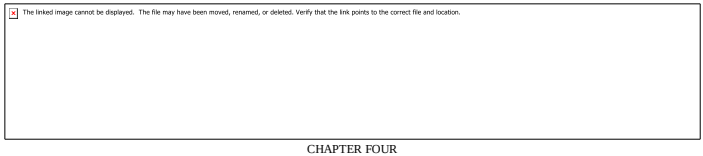
Java script date object provides several methods, they can be classified is string form, get methods and set methods. All these methods are provided in the following table.



3.8 DYNAMIC HTML

DHTML is combination of HTML, CSS and JavaScript. It gives pleasant appearance toweb page.

Difference between HTML and DHTML



CHAPTER FOUR

eXtensible Markup Language

4.1 Introduction to XML

XML was designed to describe data. HTML was designed to display data.

What You Should Already Know

Before you continue you should have a basic understanding of the following:

- HTML
- JavaScript

If you want to study these subjects first, find the tutorials on our Home page.

What is XML?

- XML stands for EXtensible Markup Language
- XML is a markup language much like HTML
- XML was designed to describe data, not to display data
- XML tags are not predefined. You must define your own tags
- XML is designed to be self-descriptive
- XML is a W3C Recommendation

The Difference Between XML and HTML

XML is not a replacement for HTML.

XML and HTML were designed with different goals:

- XML was designed to describe data, with focus on what data is
- HTML was designed to display data, with focus on how data looks

HTML is about displaying information, while XML is about carrying information.

XML Does Not DO Anything

Maybe it is a little hard to understand, but XML does not DO anything. The following example is a note to Tove, from Jani, stored as XML:

<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>

The note above is quite self descriptive. It has sender and receiver information, it also has a heading and a message body.

But still, this XML document does not DO anything. It is just information wrapped in tags. Someone must write a piece of software to send, receive or display it.

With XML You Invent Your Own Tags

The tags in the example above (like <to> and <from>) are not defined in any XML standard. These tags are "invented" by the author of the XML document.

That is because the XML language has no predefined tags.

The tags used in HTML are predefined. HTML documents can only use tags defined in the HTML standard (like <p>, <h1>, etc.).

XML allows the author to define his/her own tags and his/her own document structure.

XML is Not a Replacement for HTML

XML is a complement to HTML.

It is important to understand that XML is not a replacement for HTML. In most web applications, XML is used to describe data, while HTML is used to format and display the data.

My best description of XML is this:

XML is a software- and hardware-independent tool for carrying information.

4.2 How Can XML beUsed?

XML is used in many aspects of web development, often to simplify data storage and sharing.

XML Separates Data from HTML

If you need to display dynamic data in your HTML document, it will take a lot of work to edit the HTML each time the data changes.

With XML, data can be stored in separate XML files. This way you can concentrate on using HTML/CSS for display and layout, and be sure that changes in the underlying data will not require any changes to the HTML.

With a few lines of JavaScript code, you can read an external XML file and update the data content of your web page.

XML Simplifies Data Sharing

In the real world, computer systems and databases contain data in incompatible formats.

XML data is stored in plain text format. This provides a software- and hardware-independent way of storing data.

This makes it much easier to create data that can be shared by different applications.

XML Simplifies Data Transport

One of the most time-consuming challenges for developers is to exchange data between incompatible systems over the Internet.

Exchanging data as XML greatly reduces this complexity, since the data can be read by different incompatible applications.

XML Simplifies Platform Changes

Upgrading to new systems (hardware or software platforms), is always time consuming. Large amounts of data must be converted and incompatible data is often lost.

XML data is stored in text format. This makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.

XML Makes Your Data More Available

Different applications can access your data, not only in HTML pages, but also from XML data sources.

With XML, your data can be available to all kinds of "reading machines" (Handheld computers, voice machines, news feeds, etc.), and make it more available for blind people, or people with other disabilities.

Internet Languages Written in XML

Several Internet languages are written in XML. Here are some examples:

- XHTML
- XML Schema
- SVG
- WSDL RSS

4.3 XML Tree

An Example XML Document

XML documents use a self-describing and simple syntax:

<?xml version="1.0" encoding="UTF-8"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don’t forget me this weekend!</body>
</note>

The first line is the XML declaration. It defines the XML version (1.0).

The next line describes the **root element** of the document (like saying: "this document is a note"):

<note>

The next 4 lines describe 4 **child elements** of the root (to, from, heading, and body):

<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don’t forget me this weekend!</body>

And finally the last line defines the end of the root element:

</note>

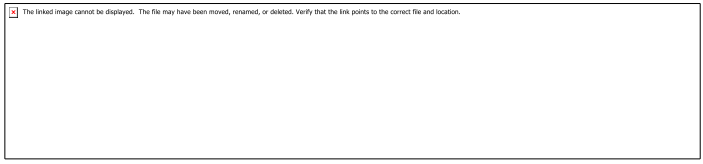


Fig 4.3.1 XML Document

You can assume, from this example, that the XML document contains a note to Tove from Jani. Don't you agree that XML is pretty self-descriptive?

XML Documents Form a Tree Structure

XML documents must contain a **root element**. This element is "the parent" of all other elements.

The elements in an XML document form a document tree. The tree starts at the root and branches to the lowest level of the tree.

All elements can have sub elements (child elements):

<root>

<child>

<subchild> </subchild>

</child>

</root>

The terms parent, child, and sibling are used to describe the relationships between elements. Parent elements have children. Children on the same level are called siblings (brothers or sisters).

All elements can have text content and attributes (just like in HTML).

Example:

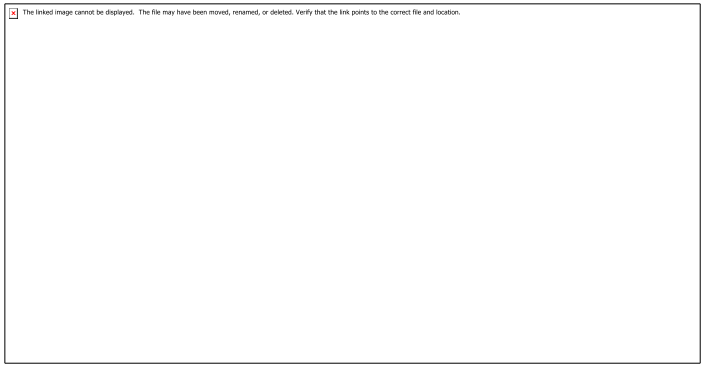


Fig 4.3.2 XML Book

The image above represents one book in the XML below:

```
<bookstore>
<book category="COOKING">
<title lang="en">Everyday Italian</title>
<author>Giada De Laurentiis</author>
<year>2005</year>
<price>30.00</price>
</book>
<book category="CHILDREN">
<title lang="en">Harry Potter</title>
<author>J K. Rowling</author>
<year>2005</year>
<price>29.99</price>
</book>
<book category="WEB">
<title lang="en">Learning XML</title>
<author>Erik T. Ray</author>
<year>2003</year>
<price>39.95</price>
</book>
</bookstore>
```

The root element in the example is <bookstore>. All <book> elements in the document are contained within <bookstore>.

The <book> element has 4 children: <title>,< author>, <year>, <price>.

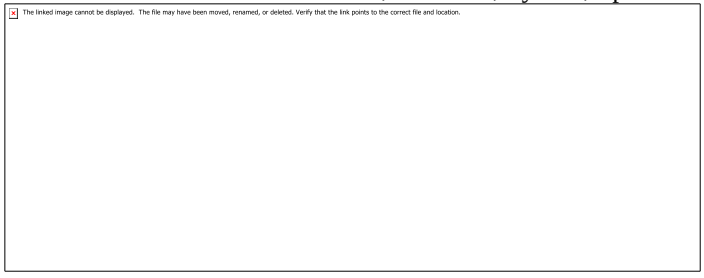


Fig 4.3.3 XML Book Store

4.4 XML Syntax Rules

The syntax rules of XML are very simple and logical. The rules are easy to learn, and easy to use.

All XML Elements Must Have a Closing Tag

In HTML, some elements do not have to have a closing tag:

This is a paragraph.

In XML, it is illegal to omit the closing tag. All elements **must** have a closing tag:

This is a paragraph.

Note: You might have noticed from the previous example that the XML declaration did not have a closing tag. This is not an error. The declaration is not a part of the XML document itself, and it has no closing tag.

XML Tags are Case Sensitive

XML tags are case sensitive. The tag <Letter> is different from the tag <letter>. Opening and closing tags must be written with the same case:

<Message>This is incorrect</message>

<message>This is correct</message>

Note: "Opening and closing tags" are often referred to as "Start and end tags". Use whatever you prefer. It is exactly the same thing.

XML Elements Must be Properly Nested

In HTML, you might see improperly nested elements:

This text is bold and italic

In XML, all elements **must** be properly nested within each other:

This text is bold and italic

In the example above, "Properly nested" simply means that since the element is opened inside the element, it must be closed inside the element.

XML Documents Must Have a Root Element

XML documents must contain one element that is the **parent** of all other elements. This element is called the **root** element.

<root>

<child>
<subchild> </subchild>
</child>
</root>

XML Attribute Values Must be Quoted

XML elements can have attributes in name/value pairs just like in HTML. In XML, the attribute values must always be quoted.

INCORRECT:

<note date=12/11/2007>
<to>Tove</to>
<from>Jani</from>
</note>

CORRECT:

<note date="12/11/2007">
<to>Tove</to>
<from>Jani</from>
</note>

The error in the first document is that the date attribute in the note element is not quoted.

Entity References

Some characters have a special meaning in XML.

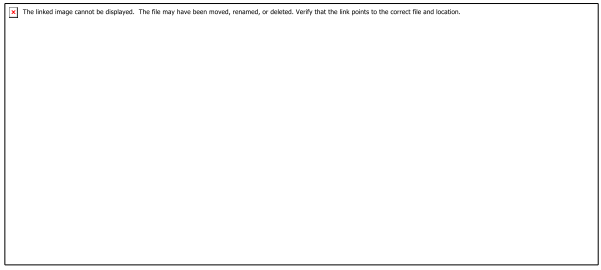
If you place a character like "<" inside an XML element, it will generate an error because the parser interprets it as the start of a new element.

This will generate an XML error:

<message>if salary < 1000 then</message>

To avoid this error, replace the "<" character with an **entity reference**:

<message>if salary < 1000 then</message> There are 5 pre-defined entity references in XML:



Note: Only the characters "<" and "&" are strictly illegal in XML. The greater than character is legal, but it is a good habit to replace it.

Comments in XML

The syntax for writing comments in XML is similar to that of HTML.

<!-- This is a comment -->

White-space is Preserved in XML

XML does not truncate multiple white-spaces in a document (while HTML truncates multiple white-spaces to one single white-space):

XML:

Hello Tove

HTML:

Hello Tove

XML Stores New Line as LF

Windows applications store a new line as: carriage return and line feed (CR+LF). Unix and Mac OSX uses LF.

Old Mac systems uses CR. XML stores a new line as LF.

Well Formed XML

XML documents that conform to the syntax rules above are said to be "Well Formed" XML documents.

4.5 XML Elements

An XML document contains XML Elements.

What is an XML Element?

An XML element is everything from (including) the element's start tag to (including) the element's end tag.

An element can contain:

- other elements
- text
- attributes
- or a mix of all of the above...

```
<bookstore>
<book category="CHILDREN">
<title>Harry Potter</title>
<author>J K. Rowling</author>
<year>2005</year>
<price>29.99</price>
</book>
<book category="WEB">
<title>Learning XML</title>
<author>Erik T. Ray</author>
<year>2003</year>
<price>39.95</price>
</book>
</bookstore>
```

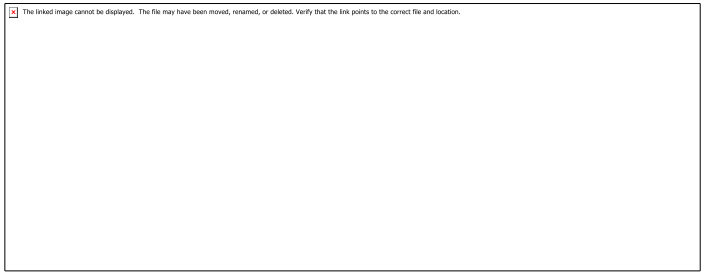


Fig 4.3.4 XML Elements

In the example above, <bookstore> and <book> have **element contents**, because they contain other elements. <book> also has an **attribute** (category="CHILDREN"). <title>, <author>, <year>, and <price> have **text content** because they contain text.

Empty XML Elements

An element with no content is said to be empty.

In XML, you can indicate an empty element like this:

<element></element>

or you can use an empty tag, like this (this sort of element syntax is called self-closing):

<element />

The two forms above produce identical results in an XML parser.

Note: Empty elements do not have any content, but they can have attributes!

XML Naming Rules

XML elements must follow these naming rules:

- Element names are case-sensitive
- Element names must start with a letter or underscore
- Element names cannot start with the letters xml (or XML, or Xml, etc)
- Element names can contain letters, digits, hyphens, underscores, and periods
- Element names cannot contain spaces

Any name can be used, no words are reserved (except xml).

Best Naming Practices

Create descriptive names, like this: <person>, <firstname>, <lastname>.

Create short and simple names, like this: <book_title> not like this: <the_title_of_the_book>.

Avoid "-". If you name something "first-name", some software may think you want to subtract "name" from "first".

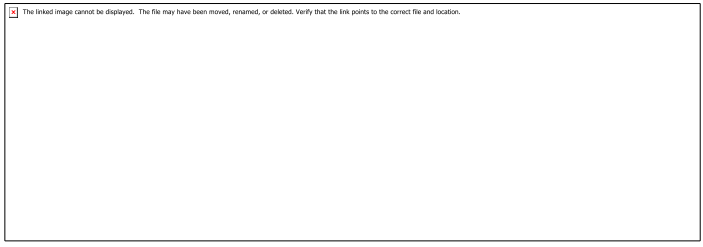
Avoid ".". If you name something "first.name", some software may think that "name" is a property of the object "first".

Avoid ":". Colons are reserved for namespaces (more later).

Non-English letters like éòá are perfectly legal in XML, but watch out for problems if your software doesn't support them.

Naming Styles

There are no naming styles defined for XML elements. But here are some commonly used:



If you choose a naming style, it is good to be consistent!

XML documents often have a corresponding database. A good practice is to use the naming rules of your database for the elements in the XML documents.

XML Elements are Extensible

XML elements can be extended to carry more information.

Look at the following XML example:

<note>
<to>Tove</to>
<from>Jani</from>
<body>Don't forget me this weekend!</body>
</note>

Let's imagine that we created an application that extracted the <to>, <from>, and <body> elements from the XML document to produce this output:

Imagine that the author of the XML document added some extra information to it:

<note>
<date>2008-01-10</date>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>

Should the application break or crash?

No. The application should still be able to find the <to>, <from>, and <body> elements in the XML document and produce the same output.

One of the beauties of XML, is that it can be extended without breaking applications.

4.6 XML Attributes

XML elements can have attributes, just like HTML. Attributes provide additional information about an element.

XML Attributes

In HTML, attributes provide additional information about elements:

<img src="computer.gif">

<a href="demo.asp">

Attributes often provide information that is not a part of the data. In the example below, the file type is irrelevant to the data, but can be important to the software that wants to manipulate the element:

<file type="gif">computer.gif</file>

XML Attributes Must be Quoted

Attribute values must always be quoted. Either single or double quotes can be used. For a person's gender, the person element can be written like this:

<person gender="female">

or like this:

<person gender='female'>

If the attribute value itself contains double quotes you can use single quotes, like in this example:

<gangster name='George "Shotgun" Ziegler'>

or you can use character entities:

<gangster name="George "Shotgun" Ziegler">

XML Elements vs. Attributes

Take a look at these examples:

<person gender="female">

<firstname>Anna</firstname>

<lastname>Smith</lastname>

</person>

<person>

<gender>female</gender>

<firstname>Anna</firstname>

<lastname>Smith</lastname>
</person>

In the first example gender is an attribute. In the last, gender is an element. Both examples provide the same information.

There are no rules about when to use attributes or when to use elements. Attributes are handy in HTML. In XML my advice is to avoid them. Use elements instead.

The Best Way

The following three XML documents contain exactly the same information: A date attribute is used in the first example:

<note date="2008-01-10">
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>

A date element is used in the second example:

<note>
<date>2008-01-10</date>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>

An expanded date element is used in the third:

<note>
<date>
<year>2008</year>
<month>01</month>
<day>10</day>
</date>
<to>Tove</to>
<from>Jani</from>

<heading>Reminder</heading>
<body>Don’t forget me this weekend!</body>
</note>

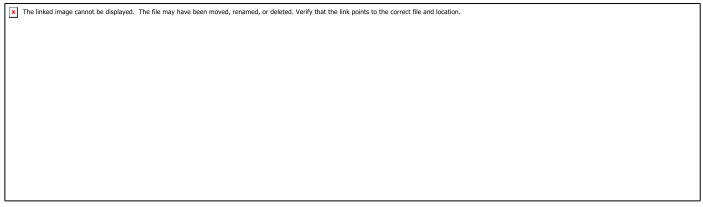


Fig 4.6.1 XML Attributes

Avoid XML Attributes?

Some of the problems with using attributes are:

- attributes cannot contain multiple values (elements can)
- attributes cannot contain tree structures (elements can)
- attributes are not easily expandable (for future changes)

Attributes are difficult to read and maintain. Use elements for data. Use attributes for information that is not relevant to the data.

Don’t end up like this:

<note day="10" month="01" year="2008" to="Tove" from="Jani" heading="Reminder" body="Don’t forget me this weekend!">
</note>

XML Attributes for Metadata

Sometimes ID references are assigned to elements. These IDs can be used to identify XML elements in much the same way as the id attribute in HTML. This example demonstrates this:

<messages>
<note id="501">
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don’t forget me this weekend!</body>
</note>
<note id="502">

<to>Jani</to>
<from>Tove</from>
<heading>Re: Reminder</heading>
<body>I will not</body>
</note>
</messages>

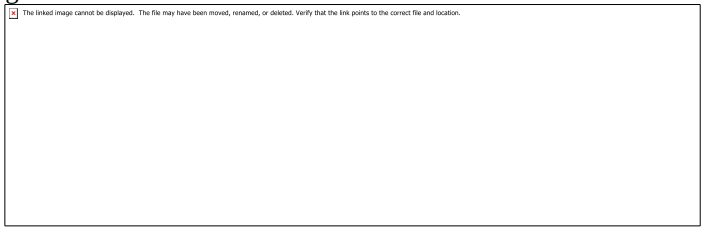


Fig 4.6.2 XML Attributes for Meta Data

The id attributes above are for identifying the different notes. It is not a part of the note itself.

What I'm trying to say here is that metadata (data about data) should be stored as attributes, and the data itself should be stored as elements.

4.7 XML Namespaces

XML Namespaces provide a method to avoid element name conflicts.

Name Conflicts

In XML, element names are defined by the developer. This often results in a conflict when trying to mix XML documents from different XML applications.

This XML carries HTML table information:

<table>
<tr>
<td>Apples</td>
<td>Bananas</td>
</tr>
</table>

This XML carries information about a table (a piece of furniture):

<table>
<name>African Coffee Table</name>
<width>80</width>

<length>120</length>
</table>

If these XML fragments were added together, there would be a name conflict. Both contain a

<table> element, but the elements have different content and meaning.

A user or an XML application will not know how to handle these differences.

Solving the Name Conflict Using a Prefix

Name conflicts in XML can easily be avoided using a name prefix.

This XML carries information about an HTML table, and a piece of furniture:

<h:table>
<h:tr>
<h:td>Apples</h:td>
<h:td>Bananas</h:td>
</h:tr>
</h:table>
<f:table>
<f:name>African Coffee Table</f:name>
<f:width>80</f:width>
<f:length>120</f:length>
</f:table>

In the example above, there will be no conflict because the two <table> elements have different names.

XML Namespaces - The xmlns Attribute

When using prefixes in XML, a so-called **namespace** for the prefix must be defined. The namespace is defined by the **xmlns attribute** in the start tag of an element.

The namespace declaration has the following syntax. xmlns:*prefix*="*URI*".

<root>
<h:table xmlns:h="http://www.w3.org/TR/html4/">
<h:tr>
<h:td>Apples</h:td>
<h:td>Bananas</h:td>
</h:tr>

</h:table>
<f:table xmlns:f="http://www.w3schools.com/furniture">
<f:name>African Coffee Table</f:name>
<f:width>80</f:width>
<f:length>120</f:length>
</f:table>
</root>

In the example above, the xmlns attribute in the <table> tag give the h: and f: prefixes a qualified namespace.

When a namespace is defined for an element, all child elements with the same prefix are associated with the same namespace.

Namespaces can be declared in the elements where they are used or in the XML root element:

<root xmlns:h="http://www.w3.org/TR/html4/" xmlns:f="http://www.w3schools.com/furniture">
<h:table>
<h:tr>
<h:td>Apples</h:td>
<h:td>Bananas</h:td>
</h:tr>
</h:table>
<f:table>
<f:name>African Coffee Table</f:name>
<f:width>80</f:width>
<f:length>120</f:length>
</f:table>
</root>

Note: The namespace URI is not used by the parser to look up information.

The purpose is to give the namespace a unique name. However, often companies use the namespace as a pointer to a web page containing namespace information.

Uniform Resource Identifier (URI)

A **Uniform Resource Identifier** (URI) is a string of characters which identifies an Internet Resource.

The most common URI is the **Uniform Resource Locator** (URL) which identifies an Internet domain address. Another, not so common type of URI is the **Universal Resource Name** (URN).

In our examples we will only use URLs.

Default Namespaces

Defining a default namespace for an element saves us from using prefixes in all the child elements. It has the following syntax:

xmlns="*namespaceURI*"

This XML carries HTML table information:

<table xmlns="http://www.w3.org/TR/html4/">
<tr>
<td>Apples</td>
<td>Bananas</td>
</tr>
</table>

This XML carries information about a piece of furniture:

<table xmlns="http://www.w3schools.com/furniture">
<name>African Coffee Table</name>
<width>80</width>
<length>120</length>
</table>

Namespaces in Real Use

XSLT is an XML language that can be used to transform XML documents into other formats, like HTML.

In the XSLT document below, you can see that most of the tags are HTML tags.

The tags that are not HTML tags have the prefix xsl, identified by the namespace xmlns:xsl="http://www.w3.org/1999/XSL/Transform":

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>

```
<h2>My CD Collection</h2>
<table border="1">
<tr>
<th style="text-align:left">Title</th>
<th style="text-align:left">Artist</th>
</tr>
<xsl:for-each select="catalog/cd">
<tr>
<td><xsl:value-of select="title"/></td>
<td><xsl:value-of select="artist"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

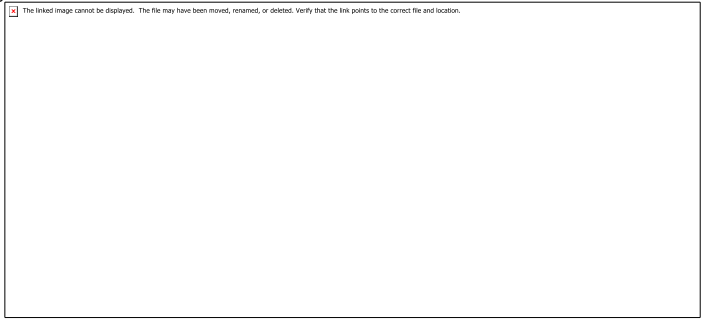


Fig 4.7.1 XML Namespaces

If you want to learn more about XSLT, please read our XSLT Tutorial.

4.8 XML Encoding

XML documents can contain international characters, like Norwegian æøå, or French êèé. To avoid errors, you should specify the encoding used, or save your XML files as UTF-8.

Character Encoding

Character encoding defines a unique binary code for each different character used in a document.

In computer terms, character encoding are also called character set, character map, code set, and code page.

The Unicode Consortium

The Unicode Consortium develops the Unicode Standard. Their goal is to replace the existing character sets with its standard Unicode Transformation Format (UTF).

The Unicode Standard has become a success and is implemented in HTML, XML, Java, JavaScript, E-mail, ASP, PHP, etc. The Unicode standard is also supported in many operating systems and all modern browsers.

The Unicode Consortium cooperates with the leading standards development organizations, like ISO, W3C, and ECMA.

The Unicode Character Sets

Unicode can be implemented by different character sets. The most commonly used encodings are UTF-8 and UTF-16.

UTF-8 uses 1 byte (8-bits) to represent basic Latin characters, and two, three, or four bytes for the rest.

UTF-16 uses 2 bytes (16 bits) for most characters, and four bytes for the rest.

UTF-8 = The Web Standard

UTF-8 is the standard character encoding on the web.

UTF-8 is the default character encoding for HTML5, CSS, JavaScript, PHP, SQL, and XML.

XML Encoding

The first line in an XML document is called the **prolog**:

<?xml version="1.0"?>

The prolog is optional. Normally it contains the XML version number.

It can also contain information about the encoding used in the document. This prolog specifies UTF-8 encoding:

<?xml version="1.0" encoding="UTF-8"?>

The XML standard states that all XML software must understand both UTF-8 and UTF-16. UTF-8 is the default for documents without encoding information.

In addition, most XML software systems understand encodings like ISO-8859-1, Windows- 1252, and ASCII.

XML Errors

Most often, XML documents are created on one computer, uploaded to a server on a second computer, and displayed by a browser on a third computer.

If the encoding is not correctly interpreted by all the three computers, the browser might display meaningless text, or you might get an error message.

For high quality XML documents, UTF-8 encoding is the best to use. UTF-8 covers international characters, and it is also the default, if no encoding is declared.

4.9 Displaying XML

Raw XML files can be viewed in all major browsers. Don't expect XML files to be displayed as HTML pages.

Viewing XML Files

<?xml version="1.0" encoding="UTF-8"?>

- <note>

<to>**Tove**</to>

<from>**Jani**</from>

<heading>**Reminder**</heading>

<body>**Don't forget me this weekend!**</body>

</note>

Look at the XML file above in your browser: note.xml

Notice that an XML document will be displayed with color-coded root and child elements. A plus (+) or minus sign (-) to the left of the elements can be clicked to expand or collapse the element structure. To view the raw XML source (without the + and - signs), select "View Page Source" or "View Source" from the browser menu.

Note: In Safari, only the element text will be displayed. To view the raw XML, you must right click the page and select "View Source".

Viewing an Invalid XML File

If an erroneous XML file is opened, some browsers report the error, and some only display it incorrectly.

Try to open the following XML file in Chrome, IE, Firefox, Opera, and Safari: note_error.xml.

Displaying XML Files with CSS?

Below is an example of how to use CSS to format an XML document.

We can use an XML file like cd_catalog.xmland a style sheet like cd_catalog.css

RESULT: The CD catalog formatted with the CSS file

Below is a fraction of the XML file. The second line links the XML file to the CSS file:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="cd_catalog.css"?>
<CATALOG>
<CD>
<TITLE>Empire Burlesque</TITLE>
<ARTIST>Bob Dylan</ARTIST>
<COUNTRY>USA</COUNTRY>
<COMPANY>Columbia</COMPANY>
<PRICE>10.90</PRICE>
<YEAR>1985</YEAR>
</CD>
<CD>
<TITLE>Hide your heart</TITLE>
<ARTIST>Bonnie Tyler</ARTIST>
<COUNTRY>UK</COUNTRY>
<COMPANY>CBS Records</COMPANY>
<PRICE>9.90</PRICE>
<YEAR>1988</YEAR>
</CD>
.
.
.
</CATALOG>
```

Formatting XML with CSS is not recommended. Use JavaScript or XSLT instead.

4.10 XML Document Types

An XML document with correct syntax is called "Well Formed".

A "Valid" XML document must also conform to a document type definition.

Well Formed XML Documents

An XML document with correct syntax is "Well Formed". The syntax rules were described in the previous chapters:

- XML documents must have a root element
- XML elements must have a closing tag
- XML tags are case sensitive
- XML elements must be properly nested
- XML attribute values must be quoted

<?xml version="1.0" encoding="UTF-8"?>

<note>

<to>Tove</to>

<from>Jani</from>

<heading>Reminder</heading>

<body>Don't forget me this weekend!</body>

</note>

An XML Validator

To help you check the syntax of your XML files, we have created an XML validatorto syntax- check your XML.

Valid XML Documents

A "valid" XML document is not the same as a "well formed" XML document.

A "valid" XML document must be well formed. In addition it must conform to a document type definition.

Rules that defines the legal elements and attributes for XML documents are called Document Type Definitions (DTD) or XML Schemas.

There are two different document type definitions that can be used with XML:

- DTD - The original Document Type Definition
- XML Schema - An XML-based alternative to DTD

When to Use a DTD/Schema?

With a DTD, independent groups of people can agree to use a standard DTD for interchanging data.

Your application can use a standard DTD to verify that the data you receive from the outside world is valid.

You can also use a DTD to verify your own data.

When to NOT to Use a DTD/Schema?

XML does not require a DTD/Schema.

When you are experimenting with XML, or when you are working with small XML files, creating DTDs may be a waste of time.

If you develop applications, wait until the specification is stable before you add a document definition. Otherwise, your software might stop working because of validation errors.

4.11 XML Validator

Errors in XML documents will stop your XML applications.

The W3C XML specification states that a program should stop processing an XML document if it finds an error. The reason is that XML software should be small, fast, and compatible.

HTML browsers will display HTML documents with errors (like missing end tags).

With XML, errors are not allowed.

Syntax-Check Your XML

To help you syntax-check your XML, we have created an XML validator.

Paste your XML into the text area below, and syntax-check it by clicking the "Validate" button.

<?xml version="1.0" encoding=

<note>

<to>Tove</to>

<from>Jani</from>

<heading>Reminder</heading>

<body>Don’t forget me this wee

</note>

Syntax-Check an XML File

You can syntax-check an XML file by typing the URL of the file into the input field below, and then click the "Validate" button:

If you get "Access Denied" or "Network Error", it is because your browser does not allow file access across domains.

The file "note_error.xml" demonstrates your browsers error handling. If you want to see an error- free message, substitute the "note_error.xml" with "cd_catalog.xml".

4.12 XML Schema

An XML Schema describes the structure of an XML document, just like a DTD. An XML document with correct syntax is called "Well Formed".

An XML document validated against an XML Schema is both "Well Formed" and "Valid".

XML Schema

XML Schema is an XML-based alternative to DTD:

<xs:element name="note">
<xs:complexType>
<xs:sequence>
<xs:element name="to" type="xs:string"/>
<xs:element name="from" type="xs:string"/>
<xs:element name="heading" type="xs:string"/>
<xs:element name="body" type="xs:string"/>
</xs:sequence>
</xs:complexType>
</xs:element>

The Schema above is interpreted like this:

- <xs:element name="note"> defines the element called "note"
- <xs:complexType> the "note" element is a complex type
- <xs:sequence> the complex type is a sequence of elements
- <xs:element name="to" type="xs:string"> the element "to" is of type string (text)
- <xs:element name="from" type="xs:string"> the element "from" is of type string
- <xs:element name="heading" type="xs:string"> the element "heading" is of type string

- <xs:element name="body" type="xs:string"> the element "body" is of type string

Everything is wrapped in "Well Formed" XML.

XML Schemas are More Powerful than DTD

- XML Schemas are written in XML
- XML Schemas are extensible to additions
- XML Schemas support data types
- XML Schemas support namespaces

Why Use an XML Schema?

With XML Schema, your XML files can carry a description of its own format.

With XML Schema, independent groups of people can agree on a standard for interchanging data.

With XML Schema, you can verify data.

XML Schemas Support Data Types

One of the greatest strength of XML Schemas is the support for data types:

- It is easier to describe document content
- It is easier to define restrictions on data
- It is easier to validate the correctness of data
- It is easier to convert data between different data types

XML Schemas use XML Syntax

Another great strength about XML Schemas is that they are written in XML:

- You don't have to learn a new language
- You can use your XML editor to edit your Schema files
- You can use your XML parser to parse your Schema files
- You can manipulate your Schemas with the XML DOM
- You can transform your Schemas with XSLT

CHAPTER FIVE

PHP & MySQL

1.PHP

5.1.1 What is PHP?

- PHP stands for **P**HP: **H**ypertext **P**reprocessor
- PHP is a widely-used, open source scripting language
- PHP scripts are executed on the server

- PHP is free to download and use

5.1.2 What is a PHP File?

- PHP files can contain text, HTML, JavaScript code, and PHP code
- PHP code are executed on the server, and the result is returned to the browser as plain HTML
- PHP files have a default file extension of ".php"

5.1.3 What Can PHP Do?

- PHP can generate dynamic page content
- PHP can create, open, read, write, and close files on the server
- PHP can collect form data
- PHP can send and receive cookies
- PHP can add, delete, modify data in your database
- PHP can restrict users to access some pages on your website
- PHP can encrypt data

With PHP you are not limited to output HTML. You can output images, PDF files, and even Flash movies. You can also output any text, such as XHTML and XML.

5.1.4 Why PHP?

- PHP runs on different platforms (Windows, Linux, Unix, Mac OS X, etc.)
- PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- PHP has support for a wide range of databases
- PHP is free. Download it from the official PHP resource: www.php.net
- PHP is easy to learn and runs efficiently on the server side

Install a web server on your own PC, and then install PHP and MySQL Find a web host with PHP and MySQL support What Do I Need? To start using PHP, you can:

5.1.5 What Do I Need?

To start using PHP, you can:

- Find a web host with PHP and MySQL support
- Install a web server on your own PC, and then install PHP and MySQL

5.1.6 Use a Web Host With PHP Support

If your server has activated support for PHP you do not need to do anything.

Just create some .php files, place them in your web directory, and the server will automatically parse them for you.

You do not need to compile anything or install any extra tools. Because PHP is free, most web hosts offer PHP support.

5.2 Basic PHP Syntax

A PHP script always starts with <?php and ends with ?>. A PHP script can be placed anywhere in the document.

On servers with shorthand-support, you can start a PHP script with <? and end with ?>.

For maximum compatibility, we recommend that you use the standard form (<?php) rather than the shorthand form.

```
<?php
// PHP code goes here
?>
```

The default file extension for PHP files is ".php".

A PHP file normally contains HTML tags, and some PHP scripting code.

Below, we have an example of a simple PHP script that sends the text "Hello World!" back to the browser:

Example

```
<!DOCTYPE html>
<html>
<body>
<?php
echo "Hello World!";
?>
</body>
</html>
```

Each code line in PHP must end with a semicolon. The semicolon is a separator and is used to distinguish one set of instructions from another.

There are two basic statements to output text with PHP: echo and print.

In the example above we have used the echo statement to output the text "Hello World".

5.3 PHP Variables

As with algebra, PHP variables are used to hold values or expressions.

A variable can have a short name, like x, or a more descriptive name, like carName. Rules for PHP variable names:

- Variables in PHP starts with a $ sign, followed by the name of the variable
- The variable name must begin with a letter or the underscore character
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0- 9, and _)
- A variable name should not contain spaces
- Variable names are case sensitive (y and Y are two different variables)

5.3.1 Creating (Declaring) PHP Variables

PHP has no command for declaring a variable.

A variable is created the moment you first assign a value to it:

```
$myCar="Volvo";
```

After the execution of the statement above, the variable **myCar** will hold the value **Volvo**.

Tip: If you want to create a variable without assigning it a value, then you assign it the value of *null*.

Let's create a variable containing a string, and a variable containing a number:

```
<?php
$txt="Hello World!";
$x=16;
?>
```

Note: When you assign a text value to a variable, put quotes around the value.

5.3.2 PHP is a Loosely Typed Language

In PHP, a variable does not need to be declared before adding a value to it.

In the example above, notice that we did not have to tell PHP which data type the variable is. PHP automatically converts the variable to the correct data type, depending on its value.

In a strongly typed programming language, you have to declare (define) the type and name of the variable before using it.

5.3.3 PHP Variable Scope

The scope of a variable is the portion of the script in which the variable can be referenced. PHP has four different variable scopes:

- local
- global
- static
- parameter

5.3.3.1 Local Scope

A variable declared **within** a PHP function is local and can only be accessed within that function. (the variable has local scope):

```
<?php
$a = 5; // global scope
function myTest()
{
echo $a; // local scope
}
myTest();
?>
```

The script above will not produce any output because the echo statement refers to the local scope variable $a, which has not been assigned a value within this scope.

You can have local variables with the same name in different functions, because local variables are only recognized by the function in which they are declared.

Local variables are deleted as soon as the function is completed.

5.3.3.2 Global Scope

Global scope refers to any variable that is defined outside of any function.

Global variables can be accessed from any part of the script that is not inside a function. To access a global variable from within a function, use the **global** keyword:

```
<?php
$a = 5;
$b = 10;
function myTest()
{
```

```
global $a, $b;
$b = $a + $b;
}
myTest(); echo $b;
?>
```

The script above will output 15.

PHP also stores all global variables in an array called $GLOBALS[*index*]. Its index is the name of the variable. This array is also accessible from within functions and can be used to update global variables directly.

The example above can be rewritten as this:

```
<?php
$a = 5;
$b = 10;
function myTest()
{
$GLOBALS[‘b’] = $GLOBALS[‘a’] + $GLOBALS[‘b’];
}
myTest(); echo $b;
?>
```

5.3.3.3 Static Scope

When a function is completed, all of its variables are normally deleted. However, sometimes you want a local variable to not be deleted.

To do this, use the static keyword when you first declare the variable: static $rememberMe;

Then, each time the function is called, that variable will still have the information it contained from the last time the function was called.

Note: The variable is still local to the function.

5.4 Parameters

A parameter is a local variable whose value is passed to the function by the calling code. Parameters are declared in a parameter list as part of the function declaration:

```
function myTest($para1,$para2,...)
{
// function code
```

}

Parameters are also called arguments. We will discuss them in more detail when we talk about functions.

A string variable is used to store and manipulate text.

String Variables in PHP

String variables are used for values that contain characters.

In this chapter we are going to look at the most common functions and operators used to manipulate strings in PHP.

After we create a string we can manipulate it. A string can be used directly in a function or it can be stored in a variable.

Below, the PHP script assigns the text "Hello World" to a string variable called $txt:

```
<?php
$txt="Hello World"; echo $txt;
?>
```

The output of the code above will be:

Hello World

Now, lets try to use some different functions and operators to manipulate the string.

The Concatenation Operator

There is only one string operator in PHP.

The concatenation operator (.) is used to put two string values together.

To concatenate two string variables together, use the concatenation operator:

```
<?php
$txt1="Hello World!";
$txt2="What a nice day!"; echo $txt1 . " " . $txt2;
?>
```

The output of the code above will be:

Hello World! What a nice day!

If we look at the code above you see that we used the concatenation operator two times. This is because we had to insert a third string (a space character), to separate the two strings.

5.5 The strlen() function

The strlen() function is used to return the length of a string. Let's find the length of a string:

```
<?php
echo strlen("Hello world!");
?>
```

The output of the code above will be:

The length of a string is often used in loops or other functions, when it is important to know when the string ends. (i.e. in a loop, we would want to stop the loop after the last character in the string).

5.6 The strpos() function

The strpos() function is used to search for a character/text within a string.

If a match is found, this function will return the character position of the first match. If no match is found, it will return FALSE.

Let's see if we can find the string "world" in our string:

```
<?php
echo strpos("Hello world!","world");
?>
```

The output of the code above will be:

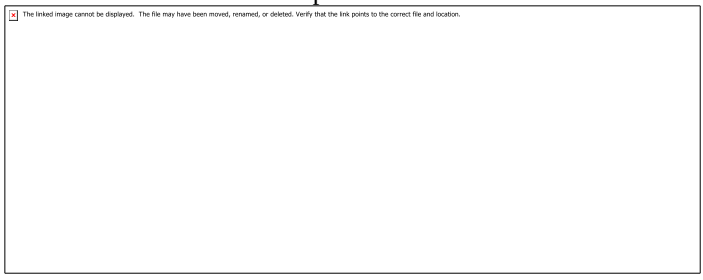
The position of the string "world" in the example above is 6. The reason that it is 6 (and not 7), is that the first character position in the string is 0, and not 1.

The assignment operator = is used to assign values to variables in PHP. The arithmetic operator + is used to add values together.

5.7 Operatore in PHP

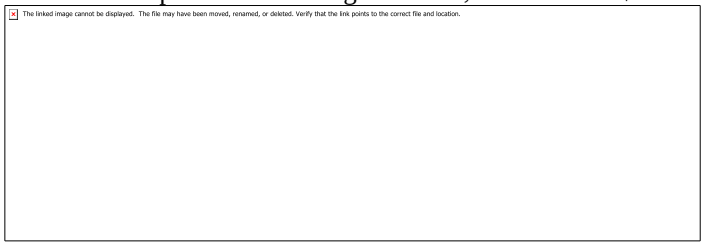
5.7.1 Arithmetic Operators

The table below lists the arithmetic operators in PHP:

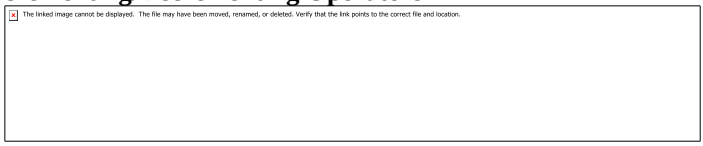


5.7.2 Assignment Operators

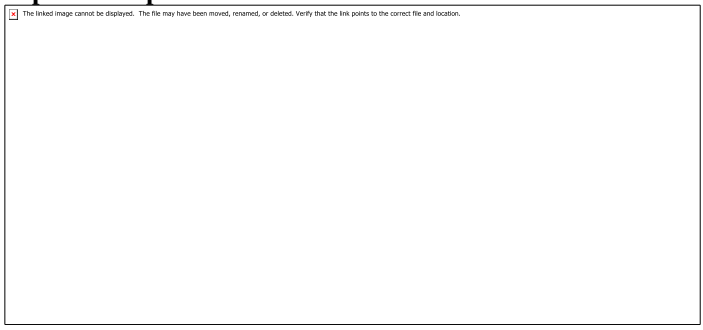
The basic assignment operator in PHP is "=". It means that the left operand gets set to the value of the expression on the right. That is, the value of "$x = 5" is 5.



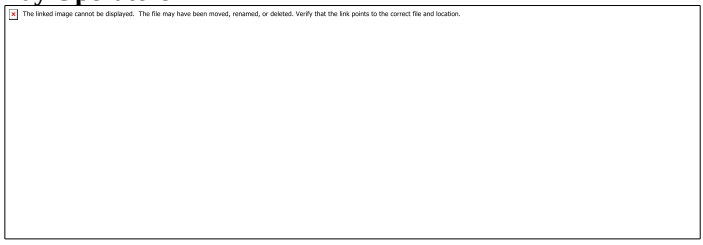
5.7.3 Incrementing/Decrementing Operators



5.7.4 Comparison Operators



5.7.5 Array Operators



5.8 Conditional Statements

Very often when you write code, you want to perform different actions for different decisions.

You can use conditional statements in your code to do this. In PHP we have the following conditional statements:

- **if statement** - use this statement to execute some code only if a specified condition is true
- **if...else statement** - use this statement to execute some code if a condition is true and another code if the condition is false
- **if...elseif else statement** - use this statement to select one of several blocks of code to be executed
- **switchstatement**- use this statement to select one of many blocks of code to be executed

5.8.1 The if Statement

Use the if statement to execute some code only if a specified condition is true.

Syntax

if (*condition*) *code to be executed if condition is true;*

The following example will output "Have a nice weekend!" if the current day is Friday:

```
<html>
<body>
<?php
$d=date("D");
if ($d=="Fri") echo "Have a nice weekend!";
?>
</body>
</html>
```

Notice that there is no ..else.. in this syntax. The code is executed **only if the specified condition is true**.

5.8.2 The if...else Statement

Use the if else statement to execute some code if a condition is true and another code if a condition is false.

Syntax

if (*condition*)
{
code to be executed if condition is true;
}
else
{
code to be executed if condition is false;
}

Example

The following example will output "Have a nice weekend!" if the current day is Friday, otherwise it will output "Have a nice day!":

```
<html>
<body>
<?php
$d=date("D");
if ($d=="Fri")
{
echo "Have a nice weekend!";
}
else
{
echo "Have a nice day!";
}
?>
</body>
</html>
```

5.8.3 The if...elseif....else Statement

Use the if....elseif...else statement to select one of several blocks of code to be executed.

Syntax

if (*condition*)
{

code to be executed if condition is true;
}
elseif (*condition*)
{
code to be executed if condition is true;
}
else
{
code to be executed if condition is false;
}

Example

The following example will output "Have a nice weekend!" if the current day is Friday, and "Have a nice Sunday!" if the current day is Sunday. Otherwise it will output "Have a nice day!":

```
<html>
<body>
<?php
$d=date("D");
if ($d=="Fri")
{
echo "Have a nice weekend!";
}
elseif ($d=="Sun")
{
echo "Have a nice Sunday!";
}
else
{
echo "Have a nice day!";
}
?>
</body>
</html>
```

5.8.4 The PHP Switch Statement

Use the switch statement to select one of many blocks of code to be executed.

Syntax

switch (*n*)
{
case *label1:*
code to be executed if n=label1;
break; case *label2:*
code to be executed if n=label2;
break; default:
code to be executed if n is different from both label1 and label2;
}

This is how it works: First we have a single expression *n* (most often a variable), that is evaluated once. The value of the expression is then compared with the values for each case in the structure. If there is a match, the block of code associated with that case is executed. Use

break to prevent the code from running into the next case automatically. The default statement is used if no match is found.

Example

```
<html>
<body>
<?php
$x=1; switch ($x)
{
case 1:
echo "Number 1"; break;
case 2:
echo "Number 2"; break;
case 3:
echo "Number 3"; break;
default:
echo "No number between 1 and 3";
}
```

?>
</body>
</html>

5.9 Array

An array stores multiple values in one single variable.

What is an Array?

A variable is a storage area holding a number or text. The problem is, a variable will hold only one value.

An array is a special variable, which can store multiple values in one single variable.

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="Saab";
$cars2="Volvo";
$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!

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 index so that it can be easily accessed. In PHP, there are three kind of arrays:

- **Numeric array** - An array with a numeric index
- **Associative array** - An array where each ID key is associated with a value
- **Multidimensional array** - An array containing one or more arrays

5.9.1 Numeric Arrays

A numeric array stores each array element with a numeric index. There are two methods to create a numeric array.

1. In the following example the index are automatically assigned (the index starts at 0):

$cars=array("Saab","Volvo","BMW","Toyota");

2. In the following example we assign the index manually:

```
$cars[0]="Saab";
$cars[1]="Volvo";
$cars[2]="BMW";
$cars[3]="Toyota";
```

Example

In the following example you access the variable values by referring to the array name and index:

```
<?php
$cars[0]="Saab";
$cars[1]="Volvo";
$cars[2]="BMW";
$cars[3]="Toyota";
echo $cars[0] . " and " . $cars[1] . " are Swedish cars.";
?>
```

The code above will output:

Saab and Volvo are Swedish cars.

5.9.2 Associative Arrays

An associative array, each ID key is associated with a value.

When storing data about specific named values, a numerical array is not always the best way to do it.

With associative arrays we can use the values as keys and assign values to them.

Example 1

In this example we use an array to assign ages to the different persons:

```
$ages = array("Peter"=>32, "Quagmire"=>30, "Joe"=>34);
```

Example 2

This example is the same as example 1, but shows a different way of creating the array:

```
$ages[‘Peter’] = "32";
$ages[‘Quagmire’] = "30";
$ages[‘Joe’] = "34";
```

The ID keys can be used in a script:

```
<?php
$ages[‘Peter’] = "32";
```

$ages[‘Quagmire’] = "30";
$ages[‘Joe’] = "34";
echo "Peter is " . $ages[‘Peter’] . " years old.";
?>
The code above will output:
Peter is 32 years old.

5.9.3 Multidimensional Arrays

In a multidimensional array, each element in the main array can also be an array. And each element in the sub-array can be an array, and so on.

Example

In this example we create a multidimensional array, with automatically assigned ID keys:

```
$families = array (
"Griffin"=>arra y (
"Peter",
"Lois", "Megan"
),
"Quagmire"=>array (
"Glenn"
),
"Brown"=>arra y ( "Cleveland"
, "Loretta", "Junior"
)
);
```

The array above would look like this if written to the output:

```
Arra y (
[Griffin] => Array (
[0] => Peter
[1] => Lois
[2] => Megan
)
[Quagmire] => Array (
[0] => Glenn
```

)
[Brown] => Array (
[0] => Cleveland
[1] => Loretta
[2] => Junior
)
)

Example 2

Lets try displaying a single value from the array above:

echo "Is " . $families['Griffin'][2]
. " a part of the Griffin family?";

The code above will output:

Is Megan a part of the Griffin family?

Loops execute a block of code a specified number of times, or while a specified condition is true.

5.10 PHP Loops

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 PHP, we have the following looping statements:

1. **while** - loops through a block of code while a specified condition is true
2. **do...while** - loops through a block of code once, and then repeats the loop as long as a specified condition is true
3. **for** - loops through a block of code a specified number of times
4. **foreach** - loops through a block of code for each element in an array

5.10.1 The while Loop

The while loop executes a block of code while a condition is true.

Syntax

while (*condition*)
{
code to be executed;
}

Example

The example below first sets a variable *i* to 1 ($i=1;).

Then, the while 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>
<?php
$i=1; while($i<=5
)
{
echo "The number is " . $i . "<br>";
$i++;
}
?>
</body>
</html>
```

Output:

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

5.10.2 The do...while Statement

The do...while statement will always execute the block of code once, it will then check the condition, and repeat the loop while the condition is true.

Syntax

```
do
{
code to be executed;
}
while (condition);
```

Example

The example below first sets a variable *i* to 1 ($i=1;).

Then, it starts the do...while loop. The loop will increment the variable *i* with 1, and then write some output. Then the condition is checked (is *i* less than, or equal to 5), and the loop will continue to run as long as *i* is less than, or equal to 5:

```
<html>
```

```
<body>
<?php
$i=1
; do
{
$i++;
echo "The number is " . $i . "<br>";
}
while ($i<=5);
?>
</body>
</html>
```

Output:

The number is 2 The number is 3 The number is 4 The number is 5 The number is 6

Loops execute a block of code a specified number of times, or while a specified condition is true.

5.10.3 The for Loop

The for loop is used when you know in advance how many times the script should run.

Syntax

for (*init; condition; increment*)
{
code to be executed;
}

Parameters:

1. *init*: Mostly used to set a counter (but can be any code to be executed once at the beginning of the loop)
2. *condition*: Evaluated for each loop iteration. If it evaluates to TRUE, the loop continues. If it evaluates to FALSE, the loop ends.
3. *increment*: Mostly used to increment a counter (but can be any code to be executed at the end of the iteration)

Note: The *init* and *increment* parameters above can be empty or have multiple expressions (separated by commas).

Example

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

```
<html>
<body>
<?php
for ($i=1; $i<=5; $i++)
{
echo "The number is " . $i . "<br>";
}
?>
</body>
</html>
```

Output:

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

5.10.4 The foreach Loop

The foreach loop is used to loop through arrays.

Syntax

```
foreach ($array as $value)
{
code to be executed;
}
```

For every loop iteration, the value of the current array element is assigned to $value (and the array pointer is moved by one) - so on the next loop iteration, you'll be looking at the next array value.

Example

The following example demonstrates a loop that will print the values of the given array:

```
<html>
```

```
<body>
<?php
$x=array("one","two","three"); foreach ($x as $value)
{
echo $value . "<br>";
}
?>
</body>
</html>
```

Output:

one two three

The real power of PHP comes from its functions. In PHP, there are more than 700 built-in functions.

5.11 PHP Built-in Functions

For a complete reference and examples of the built-in functions, please visit our PHPReference.

PHP Functions

In this chapter we will show you how to create your own functions.

To keep the script from being executed when the page loads, you can put it into a function. A function will be executed by a call to the function.

You may call a function from anywhere within a page.

5.11.1 Create a PHP Function

A function will be executed by a call to the function.

Syntax

function *functionName*()
{
code to be executed;
}

PHP function guidelines:

1. Give the function a name that reflects what the function does
2. The function name can start with a letter or underscore (not a number)

Example

A simple function that writes my name when it is called:

```
<html>
<body>
<?php
function writeName()
{
echo "Kai Jim Refsnes";
}
echo "My name is "; writeName();
?>
</body>
</html>
```

Output:

My name is Kai Jim Refsnes

5.11.2 PHP Functions - Adding parameters

To add more functionality to a function, we can add parameters. A parameter is just like a variable.

Parameters are specified after the function name, inside the parentheses.

Example 1

The following example will write different first names, but equal last name:

```
<html>
<body>
<?php
function writeName($fname)
{
echo $fname . " Refsnes.<br>";
}
echo "My name is "; writeName("Kai Jim"); echo "My sister's name is ";
writeName("Hege");
echo "My brother's name is "; writeName("Stale");
?>
</body>
</html>
```

Output:

My name is Kai Jim Refsnes.
My sister’s name is Hege Refsnes. My brother’s name is Stale Refsnes.

Example 2

The following function has two parameters:

```
<html>
<body>
<?php
function writeName($fname,$punctuation)
{
echo $fname . " Refsnes" . $punctuation . "<br>";
}
echo "My name is "; writeName("Kai Jim","."); echo "My sister’s name is ";
writeName("Hege","!");
echo "My brother’s name is "; writeName("Ståle","?");
?>
</body>
</html>
```

Output:

My name is Kai Jim Refsnes.
My sister’s name is Hege Refsnes! My brother’s name is Ståle Refsnes?

5.11.3 PHP Functions - Return values

To let a function return a value, use the return statement.

Example

```
<html>
<body>
<?php
function add($x,$y)
{
$total=$x+$y
; return
$total;
}
echo "1 + 16 = " . add(1,16);
```

?>
</body>
</html>
Output:
1 + 16 = 17
The PHP $_GET and $_POST variables are used to retrieve information from forms, like user input.

5.12 PHP Form Handling

The most important thing to notice when dealing with HTML forms and PHP is that any form element in an HTML page will **automatically** be available to your PHP scripts.

Example

The example below contains an HTML form with two input fields and a submit button:

<html>
<body>
<form action="welcome.php" method="post"> Name: <input type="text" name="fname">
Age: <input type="text" name="age">
<input type="submit">
</form>
</body>
</html>

When a user fills out the form above and clicks on the submit button, the form data is sent to a PHP file, called "welcome.php":

"welcome.php" looks like this:

<html>
<body>
Welcome <?php echo $_POST["fname"];
?>!
 You are <?php echo $_POST["age"]; ?> years old.
</body>
</html>

Output could be something like this:

Welcome John!

You are 28 years old.

The PHP $_GET and $_POST variables will be explained in the next chapters.

5.12.1 Form Validation

User input should be validated on the browser whenever possible (by client scripts). Browser validation is faster and reduces the server load.

You should consider server validation if the user input will be inserted into a database. A good way to validate a form on the server is to post the form to itself, instead of jumping to a different page. The user will then get the error messages on the same page as the form. This makes it easier to discover the error.

In PHP, the predefined $_GET variable is used to collect values in a form with method="get".

5.12.2 The $_GET Variable

The predefined $_GET variable is used to collect values in a form with method="get"

Information sent from a form with the GET method is visible to everyone (it will be displayed in the browser's address bar) and has limits on the amount of information to send.

Example

<form action="welcome.php" method="get"> Name: <input type="text" name="fname"> Age: <input type="text" name="age">

<input type="submit">

</form>

The "welcome.php" file can now use the $_GET variable to collect form data (the names of the form fields will automatically be the keys in the $_GET array):

Welcome <?php echo $_GET["fname"];

?>.
 You are <?php echo $_GET["age"]; ?> years old!

5.12.3 When to use method="get"?

When using method="get" in HTML forms, all variable names and values are displayed in the URL.

Note: This method should not be used when sending passwords or other sensitive information!

However, because the variables are displayed in the URL, it is possible to bookmark the page. This can be useful in some cases.

Note: The get method is not suitable for very large variable values. It should not be used with values exceeding 2000 characters.

In PHP, the predefined $_POST variable is used to collect values in a form with method="post".

5.12.4 The $_POST Variable

The predefined $_POST variable is used to collect values from a form sent with method="post".

Information sent from a form with the POST method is invisible to others and has no limits on the amount of information to send.

Note: However, there is an 8 MB max size for the POST method, by default (can be changed by setting the post_max_size in the php.ini file).

Example

<form action="welcome.php" method="post"> Name: <input type="text" name="fname"> Age: <input type="text" name="age">

<input type="submit">

</form>

The "welcome.php" file can now use the $_POST variable to collect form data (the names of the form fields will automatically be the keys in the $_POST array):

Welcome <?php echo $_POST["fname"];

?>!
 You are <?php echo $_POST["age"]; ?> years old.

When to use method="post"?

Information sent from a form with the POST method is invisible to others and has no limits on the amount of information to send.

However, because the variables are not displayed in the URL, it is not possible to bookmark the page.

5.12.4 The PHP $_REQUEST Variable

The predefined $_REQUEST variable contains the contents of both $_GET, $_POST, and

$_COOKIE.

The $_REQUEST variable can be used to collect form data sent with both the GET and POST methods.

Example

Welcome <?php echo $_REQUEST["fname"];
?>!
 You are <?php echo $_REQUEST["age"];
?> years old.

5.13 MySQL

What is MySQL?

1. MySQL is a database server
2. MySQL is ideal for both small and large applications
3. MySQL supports standard SQL
4. MySQL compiles on a number of platforms
5. MySQL is free to download and use

The data in MySQL is stored in database objects called tables.

A table is a collection of related data entries and it consists of columns and rows.

Databases are useful when storing information categorically. A company may have a database with the following tables: "Employees", "Products", "Customers" and "Orders".

PHP + MySQL

1. PHP combined with MySQL are cross-platform (you can develop in Windows and serve on a Unix platform)

Database Tables A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.

5.13.1 Queries

A query is a question or a request.

With MySQL, we can query a database for specific information and have a recordset returned.

Look at the following query:

SELECT LastName FROM Persons

The query above selects all the data in the "LastName" column from the "Persons" table, and will return a recordset like this:

LastName Hansen Svendson Pettersen

Create a Connection to a MySQL Database

Before you can access data in a database, you must create a connection to the database. In PHP, this is done with the mysql_connect() function.

Syntax-mysql_connect(servername,username,password);

servername-Optional. Specifies the server to connect to. Default value is "localhost:3306"

username-Optional. Specifies the username to log in with. Default value is the name ofthe user that owns the server process

password-Optional. Specifies the password to log in with. Default is ""

Example

In the following example we store the connection in a variable ($con) for later use in the script. The "die" part will be executed if the connection fails:

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
die('Could not connect: ' . mysql_error());
}
// some code
?>
```

5.13.2 Closing a Connection

The connection will be closed automatically when the script ends. To close the connection before, use the mysql_close() function:

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
die('Could not connect: ' . mysql_error());
}
// some code mysql_close($con);
?>
```

A database holds one or multiple tables.

5.13.3 Create a Database

The CREATE DATABASE statement is used to create a database in MySQL.

Syntax

CREATE DATABASE database_name

To learn more about SQL, please visit our SQL tutorial.

To get PHP to execute the statement above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection.

Example

The following example creates a database called "my_db":

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
die('Could not connect: ' . mysql_error());
}
if (mysql_query("CREATE DATABASE my_db",$con))
{
echo "Database created";
}
else
{
echo "Error creating database: " . mysql_error();
}
mysql_close($con);
?>
```

5.13.4 Create a Table

The CREATE TABLE statement is used to create a table in MySQL.

Syntax

```
CREATE TABLE
table_name ( column_name1 data_type, column_name2 data_type, column_name3 data_type,
....
)
```

To learn more about SQL, please visit our SQL tutorial.

We must add the CREATE TABLE statement to the mysql_query() function to execute the command.

Example

The following example creates a table named "Persons", with three columns. The column names will be "FirstName", "LastName" and "Age":

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
die('Could not connect: ' . mysql_error());
}
// Create database
if (mysql_query("CREATE DATABASE my_db",$con))
{
echo "Database created";
}
else
{
echo "Error creating database: " . mysql_error();
}
// Create table mysql_select_db("my_db", $con);
$sql = "CREATE TABLE
Persons (
FirstName
varchar(15), LastName varchar(15), Age int
)";
// Execute query mysql_query($sql,$con);
mysql_close($con);
?>
```

Important: A database must be selected before a table can be created. The database is selected with the mysql_select_db() function.

Note: When you create a database field of type varchar, you must specify the maximum length of the field, e.g. varchar(15).

The data type specifies what type of data the column can hold. For a complete reference of all the data types available in MySQL, go to our complete Data Types reference.

5.13.5 Primary Keys and Auto Increment Fields

Each table should have a primary key field.

A primary key is used to uniquely identify the rows in a table. Each primary key value must be unique within the table. Furthermore, the primary key field cannot be null because the database engine requires a value to locate the record.

The following example sets the personID field as the primary key field. The primary key field is often an ID number, and is often used with the AUTO_INCREMENT setting.

AUTO_INCREMENT automatically increases the value of the field by 1 each time a new record is added. To ensure that the primary key field cannot be null, we must add the NOT NULL setting to the field.

Example

```
$sql = "CREATE TABLE
Persons (
personID int NOT NULL AUTO_INCREMENT, PRIMARY
KEY(personID), FirstName varchar(15), LastName varchar(15), Age int
)";
mysql_query($sql,$con);
```

5.13.6 Insert Data Into a Database Table

The INSERT INTO statement is used to add new records to a database table.

Syntax

It is possible to write the INSERT INTO statement in two forms.

The first form doesn't specify the column names where the data will be inserted, only their values:

INSERT INTO table_name VALUES (value1, value2, value3,...)

The second form specifies both the column names and the values to be inserted:

INSERT INTO table_name (column1, column2, column3,...) VALUES (value1, value2, value3,...)

To learn more about SQL, please visit our SQL tutorial.

To get PHP to execute the statements above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection.

Example

In the previous chapter we created a table named "Persons", with three columns; "Firstname", "Lastname" and "Age". We will use the same table in this example. The following example adds two new records to the "Persons" table:

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
die('Could not connect: ' . mysql_error());
}
mysql_select_db("my_db", $con);
mysql_query("INSERT INTO Persons (FirstName, LastName, Age) VALUES ('Peter', 'Griffin',35)");
mysql_query("INSERT INTO Persons (FirstName, LastName, Age) VALUES ('Glenn', 'Quagmire',33)");
mysql_close($con);
?>
```

5.13.7 Insert Data From a Form Into a Database

Now we will create an HTML form that can be used to add new records to the "Persons" table.

Here is the HTML form:

```
<html>
<body>
<form action="insert.php" method="post"> Firstname: <input type="text" name="firstname"> Lastname: <input type="text" name="lastname"> Age: <input type="text" name="age">
<input type="submit">
</form>
</body>
</html>
```

When a user clicks the submit button in the HTML form in the example above, the form data is sent to "insert.php".

The "insert.php" file connects to a database, and retrieves the values from the form with the PHP $_POST variables.

Then, the mysql_query() function executes the INSERT INTO statement, and a new record will be added to the "Persons" table.

Here is the "insert.php" page:

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
die('Could not connect: ' . mysql_error());
}
mysql_select_db("my_db", $con);
$sql="INSERT INTO Persons (FirstName, LastName, Age) VALUES
('$_POST[firstname]','$_POST[lastname]','$_POST[age]')";
if (!mysql_query($sql,$con))
{
die('Error: ' . mysql_error());
}
echo "1 record added";
mysql_close($con);
?>
```

5.13.8 Select Data From a Database Table

The SELECT statement is used to select data from a database.

Syntax

```
SELECT
column_name(s) FROM table_name
```

To learn more about SQL, please visit our SQL tutorial.

To get PHP to execute the statement above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection.

Example

The following example selects all the data stored in the "Persons" table (The * character selects all the data in the table):

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
```

```
die('Could not connect: ' . mysql_error());
}
mysql_select_db("my_db", $con);
$result = mysql_query("SELECT * FROM Persons"); while($row =
mysql_fetch_array($result))
{
echo $row['FirstName'] . " " . $row['LastName']; echo "<br />";
}
mysql_close($con);
?>
```

The example above stores the data returned by the mysql_query() function in the $result variable.

Next, we use the mysql_fetch_array() function to return the first row from the recordset as an array. Each call to mysql_fetch_array() returns the next row in the recordset. The while loop loops through all the records in the recordset. To print the value of each row, we use the PHP

$row variable ($row['FirstName'] and $row['LastName']). The output of the code above will be:

Peter Griffin Glenn Quagmire

5.13.9 Display the Result in an HTML Table

The following example selects the same data as the example above, but will display the data in an HTML table:

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
die('Could not connect: ' . mysql_error());
}
mysql_select_db("my_db", $con);
$result = mysql_query("SELECT * FROM Persons"); echo "<table border='1'>
<tr>
<th>Firstname</th>
<th>Lastname</th>
</tr>";
```

```
while($row = mysql_fetch_array($result))
{
echo "<tr>";
echo "<td>" . $row[‘FirstName’] . "</td>"; echo "<td>" . $row[‘LastName’] .
"</td>"; echo "</tr>";
}
echo "</table>";
mysql_close($con);
?>
```

The output of the code above will be:

Firstnam e

Lastnam e

Glenn

Quagmire

Peter

Griffin

5.13.10 The WHERE clause

The WHERE clause is used to extract only those records that fulfill a specified criterion.

Syntax

```
SELECT
column_name(s) FROM table_name
WHERE column_name operator value
```

To learn more about SQL, please visit our SQL tutorial.

To get PHP to execute the statement above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection.

Example

The following example selects all rows from the "Persons" table where "FirstName=‘Peter’":

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
```

```
die('Could not connect: ' . mysql_error());
}
mysql_select_db("my_db", $con);
$result = mysql_query("SELECT * FROM Persons WHERE
FirstName='Peter'");
while($row = mysql_fetch_array($result))
{
echo $row['FirstName'] . " " . $row['LastName']; echo "<br>";
}
?>
```

The output of the code above will be:

```
Peter Griffin
```

5.13.11 The ORDER BY Keyword

The ORDER BY keyword is used to sort the data in a recordset.

The ORDER BY keyword sort the records in ascending order by default.

If you want to sort the records in a descending order, you can use the DESC keyword.

Syntax

```
SELECT
column_name(s) FROM table_name
ORDER BY column_name(s) ASC|DESC
```

To learn more about SQL, please visit our SQL tutorial.

Example

The following example selects all the data stored in the "Persons" table, and sorts the result by the "Age" column:

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
die('Could not connect: ' . mysql_error());
}
mysql_select_db("my_db", $con);
$result = mysql_query("SELECT * FROM Persons ORDER BY age");
while($row = mysql_fetch_array($result))
```

```
{
echo $row[‘FirstName’]; echo " " . $row[‘LastName’];
echo " " . $row[‘Age’]; echo "<br>";
}
mysql_close($con);
?>
```

The output of the code above will be:

Glenn Quagmire 33

Peter Griffin 35

5.13.12 Order by Two Columns

It is also possible to order by more than one column. When ordering by more than one column, the second column is only used if the values in the first column are equal:

```
SELECT
column_name(s) FROM table_name
ORDER BY column1, column2
```

Update Data In a Database

The UPDATE statement is used to update existing records in a table.

Syntax

```
UPDATE table_name SET column1=value,
column2=value2,... WHERE some_column=some_value
```

Note: Notice the WHERE clause in the UPDATE syntax. The WHERE clause specifies which record or records that should be updated. If you omit the WHERE clause, all records will be updated!

To learn more about SQL, please visit our SQL tutorial.

To get PHP to execute the statement above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection.

Example

Earlier in the tutorial we created a table named "Persons". Here is how it looks:

FirstName LastName Age

Peter Griffin 35

Glenn Quagmire 33

The following example updates some data in the "Persons" table:

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
die('Could not connect: ' . mysql_error());
}
mysql_select_db("my_db", $con); mysql_query("UPDATE Persons SET
Age=36
WHERE FirstName='Peter' AND LastName='Griffin'");
mysql_close($con);
?>
```

After the update, the "Persons" table will look like this:

FirstName LastName Age

Peter Griffin 36

Glenn Quagmire 33

5.13.13 Delete Data In a Database

The DELETE FROM statement is used to delete records from a database table.

Syntax

```
DELETE FROM table_name WHERE some_column = some_value
```

Note: Notice the WHERE clause in the DELETE syntax. The WHERE clause specifies which record or records that should be deleted. If you omit the WHERE clause, all records will be deleted!

To learn more about SQL, please visit our SQL tutorial.

To get PHP to execute the statement above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection.

Example

Look at the following "Persons" table:

FirstName LastName Age

Peter Griffin 35

Glenn Quagmire 33

The following example deletes all the records in the "Persons" table where LastName='Griffin':

```
<?php
$con = mysql_connect("localhost","peter","abc123"); if (!$con)
{
die('Could not connect: ' . mysql_error());
}
mysql_select_db("my_db", $con); mysql_query("DELETE FROM Persons
WHERE
LastName='Griffin'"); mysql_close($con);
?>
```

After the deletion, the table will look like this:

FirstName LastName Age

Glenn Quagmire 33

5.14 Create an ODBC Connection

With an ODBC connection, you can connect to any database, on any computer in your network, as long as an ODBC connection is available.

Here is how to create an ODBC connection to a MS Access Database:

1. Open the **Administrative Tools** icon in your Control Panel.
2. Double-click on the **Data Sources (ODBC)** icon inside.
3. Choose the **System DSN**tab.
4. Click on **Add** in the System DSN tab.
5. Select the **Microsoft Access Driver**. Click **Finish.**
6. In the next screen, click **Select** to locate the database.
7. Give the database a **Data Source Name(DSN)**.
8. Click **OK**.

Note that this configuration has to be done on the computer where your web site is located. If you are running Internet Information Server (IIS) on your own computer, the instructions above will work, but if your web site is located on a remote server, you have to have physical access to that server, or ask your web host to to set up a DSN for you to use.

5.14.1 Connecting to an ODBC

The odbc_connect() function is used to connect to an ODBC data source. The function takes four parameters: the data source name, username, password, and an optional cursor type.

The odbc_exec() function is used to execute an SQL statement.

Example

The following example creates a connection to a DSN called northwind, with no username and no password. It then creates an SQL and executes it:

```
$conn=odbc_connect('northwind','','');
$sql="SELECT * FROM customers";
$rs=odbc_exec($conn,$sql);
```

5.14.2 Retrieving Records

The odbc_fetch_row() function is used to return records from the result-set. This function returns true if it is able to return rows, otherwise false.

The function takes two parameters: the ODBC result identifier and an optional row number:

```
odbc_fetch_row($rs)
```

5.14.3 Retrieving Fields from a Record

The odbc_result() function is used to read fields from a record. This function takes two parameters: the ODBC result identifier and a field number or name.

The code line below returns the value of the first field from the record:

```
$compname=odbc_result($rs,1);
```

The code line below returns the value of a field called "CompanyName":

```
$compname=odbc_result($rs,"CompanyName");
```

5.14.4 Closing an ODBC Connection

The odbc_close() function is used to close an ODBC connection.

```
odbc_close($conn);
```

An ODBC Example

The following example shows how to first create a database connection, then a result-set, and then display the data in an HTML table.

```
<html>
<body>
<?php
$conn=odbc_connect('northwind','','');
if (!$conn)
{exit("Connection Failed: " . $conn);}
$sql="SELECT * FROM customers";
$rs=odbc_exec($conn,$sql);
if (!$rs)
```

```
{exit("Error in SQL");} echo "<table><tr>";
echo "<th>Companyname</th>"; echo "<th>Contactname</th></tr>"; while
(odbc_fetch_row($rs))
{
$compname=odbc_result($rs,"CompanyName");
$conname=odbc_result($rs,"ContactName"); echo "<tr><td>$compname</td>";
echo "<td>$conname</td></tr>";
}
odbc_close($conn); echo "</table>";
?>
</body>
</html>
```