1 The Way the Web Works

LEARNING OBJECTIVES

At the end of this chapter, you will be able to:

- Describe the WWW and Internet.
- List the layers in OSI model and define their functions.
- List the layers used by Internet with corresponding protocols.
- Define the meaning of protocol.
- Discuss the widely used Internet protocols for file, email, instant messaging and remote login.
- Describe the HTTP protocol used to transport the Web pages.
- Discuss the HTTP Request and Response formats.
- Differentiate between GET and POST methods.
- List the protocols used for making Internet communication secure.
- List the applications and development tools used for Web development.
- Define the role of Web browsers and list their features.
- Draw the architecture of a browser and list the function of each component.
- Define the term 'server' and list various types of software servers.
- List the various Web servers and discuss how to set up a Web server for internal and external access.
- Show the use of XAMPP server to publish the Web pages.
- List the features and applications of Web 2.0.

A basic introduction to how the WWW works within the context of the Internet with supporting protocols and applications. This first chapter aims at equipping you with a thorough grounding in what you need to know to start developing applications and pages for the Web.

To begin with there is a brief history lesson on how the Internet and in particular the World Wide Web (WWW) was developed, which explains how various aspects of the technology such as hypertext came about. As time went on, more applications were found that could be handled by the Internet, each one of these slowly becoming standardized in the way they went about communicating across the Web until there were many protocols available for different tasks. The more important protocols are explored here.

To develop applications you need to be able to interact with a server that will show your Web pages and run your applications. To communicate with this machine you need a way of uploading and talking to it. This is shown here along with some issues you may meet along the way.

One of the most important tools you will be using while developing will be the Web browser so some of its useful features, to both a user and developer, are explained. This is approached in a generic way and therefore should be equally applicable to your favorite browser. Another important software tool needed for successful delivery of Web page is Web server. There are number of Web server software available and one needs to choose from it. The Web 2.0 sites and applications are widely used and need attention.

CamScanner

1.1 History

First of all, let's start with a bit of history – how did it all begin and why? As long as information has been collected and stored there has been a need for duplication and, with that, a means for communicating it to other places. This was true when the first books and other documents were being written throughout history. When computers came on the scene it became important that information could be stored and transferred between machines so other people could access the information and maybe add to it.

Computers were initially linked together to store information in universities, defense organizations and government departments. As they did not all have a common standard to communicate by, information could not be passed between dissimilar systems. A university may have had a department where scientists could share their work and papers but it would become a problem to connect to someone else's system and transfer documents or data.

Possibly the biggest driving factor to develop interoperability between systems was the needs of the US Department of Defense. They required a communications architecture for command and control that was reliable, universal and in some way self-healing. The idea here was that if a certain center, or node, was lost in the network (for example by enemy attack) then information would not be disrupted but would find its way around it to its intended destination. In the late 1960s, the Defense Advanced Research Projects Agency (DARPA) worked with both industry and universities, starting a network to test various ideas out in this line. The network was known as ARPANET and was used to develop protocols and techniques that are now used on the Internet.

The standards that were developed in that period were passed on in the 1980s to the US Department of Defense's Defense Communications Agency. The agency became their guardian until finally, with the establishment of a more widely used Internet in the 1990s, they were passed on to the Internet Architecture Board (IAB), an independent organization.

The WWW

The WWW is part of the history of the Internet. Its inventor, Tim Berners-Lee, in 1980 was investigating how computers could store information with random links. In 1989, while working at the European Particle Physics Laboratory, he proposed the idea of a global hypertext space in which any network-accessible information could be referred to by a single 'Universal Document Identifier'. In 1990 this idea was expanded with a program called 'WorlDwidEWeb', a point and click hypertext editor running on a NeXT computer; further developments led to the browser and Web server. Several specifications were developed here including URIs, HyperText Markup Language (HTML) and HyperText Transfer Protocol (HTTP) and were in fact published on the first server to promote wide adoption and discussion.

Over the years 1991 to 1994 the load on the first Web server info.cern.ch increased by a factor of ten every year. A selection of browsers was developed for different types of computer. Initially academia and then industry started taking notice. Under pressure to define future direction, Tim Berners-Lee formed the World Wide Web Consortium (W3C) in 1994, which acted as a neutral forum where companies and organizations could go to discuss and agree on new common computer protocols.

1.2 The Internet and the WWW

The Internet works by defining an address for each resource attached to it. In the case of attached computers and some devices, this is the Internet Protocol (IP) address. Without some form of address, no link could be formed between computers, resources or systems. An example of an IP address is 158.162.131.236. The numbers break down to a country, to a specific domain, right down to a machine itself.

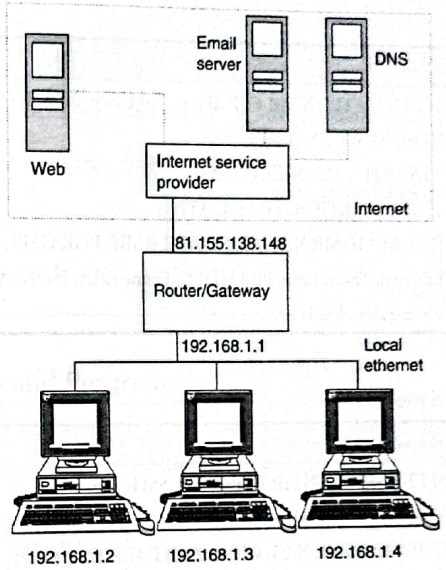


Figure 1 How computers link to the Internet.

Ordinarily, of course, we don't use numbers to find resources and machines – we use a name like mycomputer.co.uk. The name actually will have an associated number but when a name is entered, say, into a Web browser, the computer will use a special system called the Domain Name System (DNS) to look up the associated number which it will then use to get in contact with the desired server. Figure 1 shows the basic idea behind IP addressing for one particular network. In this diagram a group of computers are connected to a Local Area Network (LAN), each having a local IP address. These are connected to a gateway or router, which in turn is linked to an Internet Service Provider (ISP), usually with some type of modem. The ISP allows access to the Internet, DNS lookup services and email. The ISP also allocates an IP address for the network, which in turn can be utilized to resolve to individual machines using techniques such as Network Address Translation (NAT).

The discussion behind how the actual communication takes place deserves a book or course of its own but the basics will be described here. Any communication that occurs on the Internet uses the TCP/IP (Transport Control Protocol/Internet Protocol) suite. This is a set of communication protocols, or conventions, for dialogue between computers and devices. These protocols implement a stack, each layer of which solves problems relating to the transmission of data, as well providing service to higher layers than itself. Higher layers are logically closer to the user, dealing with more abstract data and rely on the lower layers to provide the means in which they can be physically manipulated.

Table 1 shows the Operating Systems Interconnection (OSI) reference model, which is a layered abstract description for describing communications and computer network protocols. It is roughly adhered to in the computing and networking industry and is often simplified to produce a model for the Internet as shown in Table 2. Note here that the session and presentation layers have been absorbed into the application layer.

CamScanner

Table1 OSI model

OSI Level	*Stack Layer*	*Protocol*
7	Application	HTTP, SMTP, SNMP, FTP, Telnet, SSH, Scp, NFS, RTSP
6	Presentation	XDR, ASN.1, SMB, AFP
5	Session	TLS, SSH, RPC, NetBIOS, ASP
4	Transport	TCP, UDP, RTP, SCTP, SPX, ATP
3	Network	IP, ICMP, IGMP, X.25, CLNP, ARP, RARP, BGP, OSPF, RIP, IPX, DDP
2	Data Link	Ethernet, Token Ring, PPP, HDLC, Frame Relay, ISDN, ATM, 802.11 Wi-Fi, FDDI
1	Physical	Electrical, radio, laser

Table 2 Internet layered model

OSI Level	*Stack Layer*	*Protocol*
7	Application	HTTP, SMTP, SNMP, FTP, Telnet, SSH, Scp, DNS
4	Transport	TCP, UDP, RTP, SCTP, SPX, ATP
3	Network	IP, ICMP, IGMP, X.25, CLNP, ARP, RARP, BGP, OSPF, RIP, IPX, DDP
2	Data Link	Ethernet, Token Ring, PPP, HDLC, Frame Relay, ISDN, ATM, 802.11 Wi-Fi, FDDI
1	Physical	Electrical, radio, laser

Where the OSI model was theoretical and produced at an earlier stage in the evolution of networks, the Internet model, as shown in Table 2, was produced as a practical solution to the engineering problems involved.

The *physical layer* deals with the physical characteristics of communication such as conventions about the medium used for communication (e.g. wires, fiber optic or radio links). Other related details such as connectors, channels, modulation, signal strengths, level synchronization, timing and distances involved are also included here.

The *data link layer* specifies how packets of information are transported over the physical layer, such as how it is framed or set out with special start and stop bit patterns. The *network layer* is concerned with how the packets are transferred over and between networks.

Protocols at the *transport layer* can deal with solving problems like reliability; that is, whether or not the data reached the location it was intended to and ensuring that data arrives in the correct order. Transport protocols within the TCP/IP suite also determine which application any given data is intended for. TCP is one such protocol that exists here and is said to be reliable and connection oriented. UDP (User Datagram Protocol) also exists at this level and has some interesting characteristics. For example, because it is usually used for streaming video or audio it is a 'best effort' or 'unreliable' protocol in that it does not verify that packets reach their destination; rather its concern is that they arrive on time.

Finally, the *application layer* is the layer that programs use to interface with in order to communicate across a network with other programs. Data is passed from the program in an application format and encoded into a standard protocol. Some programs are considered to run in this layer, with their associated protocols. Once the data from an application has been encoded into a standard application layer protocol it is then passed down to the next layer of the OSI model.

Checkpoint!

We now know:

- There was an initial need for information exchange between systems.
- Information exchange was the catalyst for developing networks.
- Networks require common conventions for dialogue (protocols).
- Each resource in a network needs a unique identifier (IP address).
- An IP address can be associated with a name using DNS.
- A stack of protocols exist, providing services and communication between the various levels in a system.

1.3 Protocols and Programs

To enable communications to occur between parties, there are a few aspects that have to be considered:

1. Where the communication takes place (usually a specific *port*, a kind of interface for communication over a network).
2. How it takes place.
3. The rules and conventions involved.

For the Internet, there are many ways of communicating between machines and devices, depending on what is actually required. Often there are many ways to do the same thing, which is important because the particular context that a device is in may limit what resources are available to it.

A specific task usually requires a particular form of communication. Here we look at a few contexts in which we may wish to transfer information between systems and the mechanisms available for the task.

Files

If you want a file to go from one machine to another, how would you do it? You could, for example, simply email it and pick up the email on the other machine. There may be problems with this – what if there was no email client or the file was simply too big? What other possibilities are there? You could use a disk or memory of some kind. Another way could be to use File Transfer Protocol (FTP) to load it either directly on the machine, or to a server, so it can be downloaded on to the target machine later. Let's say you want to transfer a file called myIndex.doc; a typical session may run like this:

```
>ftp mysuperserver.co.uk
Connected to mysuperserver.co.uk.
220 FTP Server ready.
Name (mysuperserver.co.uk:ralphmoseley): ralph101
331 Password required for ralph101.
Password:
230 User ralph101 logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>put myIndex.doc
```

The file will then be transferred. Notice that different files require either binary or ASCII (American Standard Code for Information Interchange, pronounced *as-key*, a character set) modes, text-based files

need to be sent using ASCII. Graphic files, such as JPEG or GIF, and word processor formats, such as MS Word doc, require binary.

When you are logged in you have a range of commands available, mainly based on UNIX commands, such as those shown in Table 3.

As you can see, there are lots of commands but there are some that are used more than others:

1. ftp – start an ftp session
2. ls – list files
3. get – download a file from the server
4. put – upload a file to the server
5. mkdir – make a directory on the server
6. cd – change to a new directory on the server
7. close – close the connection
8. open – open a new connection
9. bin – binary mode transfer
10. asc – ASCII text mode transfer.

Table 3 FTP command set

!	Features	mls	Proxy	Size
$	fget	mlsd	put	sndbuf
account	form	mlst	pwd	status
append	ftp	mode	quit	struct
ascii	gate	modtime	quote	sunique
bell	get	more	rate	system
binary	glob	mput	rcvbuf	tenex
bye	hash	msend	recv	throttle
case	help	newer	reget	trace
cd	idle	nlist	remopts	type
cdup	image	nmap	rename	umask
chmod	lcd	ntrans	reset	unset
close	less	open	restart	usage
cr	lpage	page	rhelp	user
debug	lpwd	passive	rmdir	verbose
delete	ls	pdir	rstatus	xferbuf
dir	macdef	pls	runique	?
disconnect	mdelete	pmlsd	send	
edit	mdir	preserve	sendport	
epsv4	mget	sendport	set	
exit	mkdir	progress	site	
		prompt		

These provide an adequate capability to allow a user to get their files from or upload them to a server. When developing Web pages or applications, there is a need for a cycle of local editing and updating of the files on the server. This can be done with an FTP connection; some editors even include the ability to transfer files online!

Concept Check

- Access a command line window and try connecting to an FTP site using the commands mentioned above – ls, put, get …

Problems with FTP

There are some possible problems when using FTP that you should be aware of. Most machines connected to the Internet have some kind of firewall in place to stop intrusion. A firewall can be part of your operating system, a piece of software you install or included on a piece of hardware on your network. A firewall may be set up to stop access either outward bound or inward toward your computer, on any of the available ports. This includes the FTP port 21. So, if your computer is having connection problems it's worth checking this; it is usually possible to allow communication on specific ports while others are blocked.

It must also be remembered that normal FTP on port 21 is not very secure and it is possible to intercept and view data that is sent, as well as any passwords. This is because information is sent as clear text without any form of encryption. If the data is sensitive or personal, other methods exist for transfer of information.

Lastly, if you use FTP to transfer files for your Web site, don't forget that any newly uploaded pages may take a while to appear to a browser due to factors such as local caching (a store to help speed up regularly viewed pages) on individual machines and servers.

Checkpoint!

We now know how to:

- Open an FTP connection to another machine.
- Do various operations such as listing the files on the remote server.
- Get single and multiple files from the server.
- Put single and multiple files on the server.
- Close the FTP connection.
- Be aware of problems concerning FTP such as security and firewalls.

Email

Email existed as long ago as 1965 on time-shared user mainframes and pre-dates the Internet itself. The use was extended to work on networks between many computers. Modern email works by a user writing their message using a mail client (or Mail User Agent, MUA); this program then uses Simple Mail Transfer Protocol (SMTP) to send the message to the local mail transfer agent (MTA), which usually exists on the user's ISP. The MTA then deciphers the email address of the recipient (who the email is to!), which takes the form of myname@myaddress, where myname is the local part and myaddress is the domain name. The MTA uses the DNS to find the appropriate mail exchange server accepting messages for that domain. Once the mail server is found, the message is sent on using SMTP, and from there it is placed using the local name (myname) to find the correct mail box.

CamScanner

To pick up the mail, a user's client retrieves it from the mail box via his MUA, probably using Post Office Protocol (POP3).

Instant Messaging

It's possible to use an Instant Messaging (IM) service to talk to a friend on another computer, so how does this work? The problem here is that the protocols and programs in use are fairly, at this point in time, non-standard in the sense that each has its own approach. One of the more popular is Internet Relay Chat (IRC), which is designed for group communication in channels (group areas where people meet) but also allows one to one communication. IRC is an open protocol that uses TCPexTCP and if desired, Secure Sockets Layer (SSL). Users connect with a client application (of which there are many, for each type of operating system), which links to a IRC server. The actual protocol is plain text; that is, it is quite possible to link to an IRC server with Telnet. However, it is easier to use a client as some character encoding can cause problems and the client makes it considerably easier with built-in commands, for example.

An interesting development on IRC were bots, automated clients that perform some duty or service, such as exercising operator privileges (controlling channels and acting, if abuse is detected). There are also bots that perform the opposite to annoy users by sending out unwanted messages to them!

Remote Machine Access

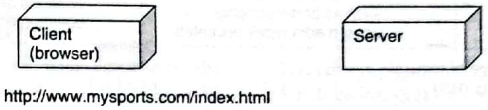
If you need to access a machine from a distance, to maybe run a program or retrieve some data, there are a few ways to do this. One way is through Telnet, which allows you to open a connection to a remote machine and issue commands as if you were actually sat at that machine. It is important to understand that the other end of the connection must be a process (or daemon) that will act as a server. Telnet uses port 23 and is largely now considered a security risk due to vulnerabilities that have manifested themselves over the years. This comes from the degree of flexibility allowed by the Telnet program and also from weaknesses within the protocol. Telnet does not encrypt any communication, so passwords are open to eavesdropping, which can be done fairly easily. It is also possible to hijack (or, take over) a Telnet session due to the lack of authentication between the parties, so it's impossible to know whether parties are who they pertain to be. A better means of remote access is to use SSH (Secure Shell), which is, like Telnet, both a program and a protocol. A typical SSH session runs like this (user typing in bold):

```
Raptor-Computer: ~ ralphmoseley$ ssh ralph@192.168.1.4
Password:[type password]
Last login: Sat Mar 12 18:50:54 2005 from 192.168.1.2
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994
The Regents of the University of California. All rights
                reserved.

FreeBSD 4.10-RELEASE (GENERIC) 0: Tue May 25 22:47:12 GMT 2004
Welcome to FreeBSD!

aphid# ls
.cshrc    .login                .rnd        server.csr
.history  .mysql_history        mbox        server.key
.klogin   .profile              server.crt
aphid# logout
Connection to 192.168.1.4 closed.
```

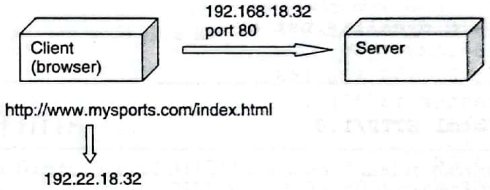

Web Pages

The main protocol used for communication between a browser and a Web server is HTTP. This protocol was designed to enable documents to be transferred but can be used with other types of data too. For Web documents the HTTP protocol works by sending commands over a TCP connection.

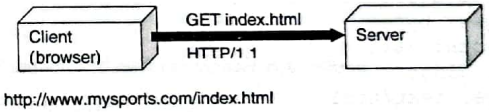
To understand how information gets passed from machine to machine we need to know how such systems can connect to each other. Generally, the main model used for the Web is client–server.



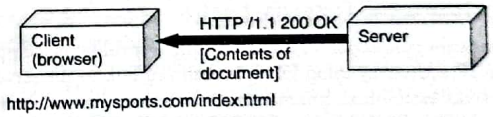
The first stage is the user typing a URL in the browser address window.



In the next stage the URL is converted to an IP address, which is then used to make a connection to the server at that location via port 80, the one used for HTTP and the Web.



Once the connection is established, the client application extracts the file name that is required from the URL and sends the request down the established connection. When received, the server looks up the request.



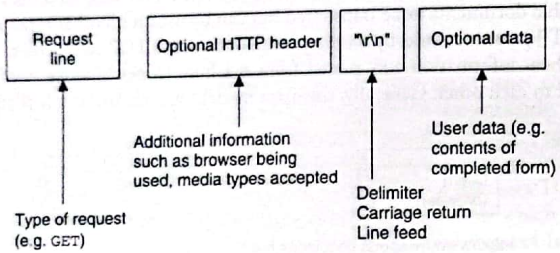
If all is well the HTTP message is sent saying that the page was found, followed by the page itself. When the page has been sent the connection is dropped.

The following HTTP command may originate from a browser to request a Web page from a server:

```
GET /index.html HTTP/1.0
```

CamScanner

This command, which is a text-based command, as above, has several fields:



In fact, it is quite easy to pretend to be a Web browser! Using a Telnet program it is possible to connect to a Web server and ask for the initial page, usually index.html or index.htm. The session may run something like this:

```
$ telnet aphid.dynalias.net 80
Trying 81.155.138.148...
Connected to 81.155.138.148.
Escape character is '^]'.
GET /index.html HTTP/1.0

HTTP/1.1 200 OK
Date: Fri, 04 Mar 2005 20:02:01 GMT
Server: Apache/1.3.29 (Unix) PHP/4.3.6 mod_ssl/2.8.16
                OpenSSL/0.9.7d

Last-Modified: Mon, 06 Sep 2004 12:43:33 GMT
ETag: "4c9003-a71-413c5b75"
Accept-Ranges: bytes
Content-Length: 2673
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<!--Web page here -->
</HTML>
Connection closed by foreign host.
```

The initial command opens the connection to the server. The first thing the computer does is translate the name of the server to an IP address by using DNS. It then can link to the server and form a temporary connection for requests. Once established, commands can be written just as a browser does. Here there is a request for a Web page, index.html, using protocol HTTP 1.0. The server then responds by saying the command was understood, followed by some other identification data and then by the Web page itself. Finally, the connection is closed as the request has been dealt with.

Lots of things can go wrong when asking for a Web page in this way! You may write the command in carefully but the page doesn't exist, in which case you will receive an error code back instead of a Web page. Some of the more common ones are:

1. 404 = Not found
2. 401 = Unauthorized
3. 500 = Internal server error
4. 501 = Not implemented.

There are quite a few more too, such as errors involving redirection services. A browser will also send other information along with the request, identifying its various properties and capabilities but not the user.

Concept Check

Using a command line window as above, type the telnet command to talk to a Web server on port 80 e.g.:

```
telnet www.yahoo.com 80
GET /index.html HTTP/1.1
[press enter twice]
```

If you type this correctly you could get a few responses:

The correct page
A redirection page
Can't find it page

All will be in HTML, as you would expect! The connection will then close (it's not a persistent connection, remember).

You also try this with some terminal programs such as PuTTY.

1.4 Review of HTTP

We know that HyperText Transfer Protocol (HTTP) is a protocol used to transfer an HTML page from Web server to Web browser upon making a request from the browser using URL. The major components of the HTTP are as follows:

1. HTTP request message.
2. HTTP response message.
3. HTTP methods.
4. HTTP status codes.

The format for the HTTP request and response message is as follows:

```
Request/Response line
Message Headers
<CR> <LF>
<Message Body>
```

First line is either request line or response line. Second line is message headers and represents one or more headers in form of header name followed by colon and value of the header. After list of headers, an empty line (<CR><LF>) is present to separate them from message body.

For a request message, the headers are implicit parameters send by the Web browser to Web server. Web browser uses them to pass some details to Web server. For example, `User-Agent` header is used to indicate the details of the browser. The message body depends on the request method. For GET method, it is empty whereas for POST it contains the user-defined values, i.e. explicit parameters in following format:

```
name1=value1&name2=value2
```

The same are send in GET method using Query String appended to URL after? as

```
http://www.myweb.com/index.html?name1=value1&name2=value2
```

CamScanner

Following are some examples of request message.

```
1. GET /index.html                          HTTP/1.1
2. GET /cgi-bin/first.cgi                   HTTP/1.1
   Accept : */*
   User-Agent : NetScape 6.2
   ...
   Connection : Keep-Alive
3. POST      /cgi-bin/reg_user.cgi          HTTP/1.1
   Accept : */*
   User-Agent : NetScape 6.2
   ...
   Connection : Keep-Alive
   name=Mansukh+Savaliya&city=Ahmedabad
```

The request line starts with method followed by path of the resource of file and version of HTTP. First example is simple that requests the index.html file using GET method. Second example requests the CGI program stored in `cgi-bin` directory on Web server using GET method. It shows some of the message headers send by browser to Web server. Accept header indicates which Multipurpose Internet Mail Extensions (MIME) data types it can accept. `User-Agent` indicates which browser it is. Third example uses the POST method with parameters passed in message body, which are passed in GET method as Query String appended to URL.

For response message, the first line shows the status of the response followed by message headers send by Web server to Web browser. Web server uses message headers to pass some details to Web browser. For example, `Content-Type` header is used to indicate the MIME type of the response. Finally, the message body contains the HTML page received as response. For example,

```
HTTP/1.1    200   OK
Content-Type : text/html
...
...
<html>
...
...
...
</html>
```

The status code 200 says response is received without any problem. Header `Content-Type` indicates that message body contains valid HTML page. The message body contains complete HTML page that was requested.

Besides GET and POST methods, HTTP has other methods such as HEAD, PUT, DELETE etc. However, GET and POST are widely used methods. The major difference between them is the way the parameters are passed. GET passes them as Query String appended to URL and POST passes them in message body. The major limitations of GET method is that it is insecure and can accommodate only limited data. For a large data to be send securely, POST is considered to be the best method.

The various status codes indicate various meaning to the browser. Status code 200 indicates everything was fine. Status code 404 indicates the requested resource was not found. Status code 500 indicates there was some internal error on the server. There is a long list of status codes having different meanings.

All the Internet protocols are described by request for comments (RFCs) and are available on the Internet. For more details on HTTP, it is advised to study appropriate RFC from the Internet.

Concept Check

Make a detail study of HTTP with request headers, response headers, various methods and meaning of various status codes. You can use various Web resources or RFCs.

Checkpoint!

We now know:

- HTTP stands for "HyperText Transfer Protocol" and used to transfer Web pages from Web server to Web browser.
- HTTP consists of HTTP Request message, HTTP Response message, HTTP methods and Status codes.
- GET method passes parameter as Query String in URL whereas POST method transfers the parameters in message body

1.5 Secure Connections

Most of the protocols that have been looked at are not secure; they use plain text to transfer data and could be viewed or tampered with at some stage. There are some protocols that make it more secure to send and receive data, which include special versions of FTP as SFTP/FTPS and HTTP as HTTPS. Other ways of actually executing and logging into a machine exist too, similar to Telnet. These include SSH and various virtual networking tools that allow a user to create a secure tunnel through to the host machine, almost as if the connection were local. Table 4 shows the various protocols, associated ports (which can be changed) and whether security is available with that particular communication mode.

SSH allows communications to take place in a secure manner over port 22, with encryption, minimizing the risk of interception or tampering. The SSH program is similar to Telnet in that it allows a user to log in and execute commands on a remote computer. It is usually included with UNIX (and its variants) as a program. A server process/daemon usually also exists to accept incoming requests. Other operating systems have similar programs available that are freeware or shareware, such as WinSCP.

To transfer files in a similar way to FTP there exists a version that uses a secure method of communication, SFTP. It is important to note that there are several protocols with this name: the one referred to here is Secure (Shell) File Transfer Protocol (another one is Simple File Transfer Protocol). SFTP can refer to the protocol or the application program involved, depending on the context it is used in. It is not simply an FTP connection run through a secure shell but a completely new protocol.

Yet another version of FTP exists as FTPS, which utilizes Secure Sockets Layer (SSL)/Transport Layer Security (TLS) to secure any connections made.

For shopping, banking and financial transactions on the Web another protocol was called for, besides the insecure and easy to intercept HTTP. This is HTTPS and it provides authentication and encryption for electronic commerce by encrypting communications using a version of SSL or TLS. You will know when you encounter it as the URL is `https://` rather than `http://`. HTTPS provides a measure of security while the data, such as credit card details, is in transit to the server. However, once at the server and possibly stored in a database, the data may still be open to attack prior to any further transmission to a credit card processor, for example.

CamScanner

Table 4 Transfer of various media and appropriate attributes

Communication	*Application*	*Protocol*	*Port*	*Security*
Web page	Browser	HTTP	80	–
Web page	Browser	HTTPS	443	Secure
Files[Binary/Text]	FTP	FTP	21	–
Files[Binary/Text]	SFTP	SFTP	22	Secure
Files[Binary/Text]	FTP	FTPS	990	Secure
Commands	Telnet	Telnet	23	–
Commands	SSH	SSH	22	Secure
Instant Messaging	IRC	IRC	194	Not usually secure

Checkpoint!

We now know:

- The importance of using the correct level of security required by a situation.
- The choices available to provide a secure alternative.
- SFTP and FTPS can be used in place of FTP.
- SSH can be used in place of Telnet.
- The SSH TCP port is 22.

1.6 Applications and Development Tools

For example, you can use FTP from a console window in Windows or a UNIX-based system and it will allow you a basic level of ability to move, copy or manipulate files. It is also possible to use a far more complex program that enhances capabilities in some sense. This may all be presented in a graphical manner rather than text driven so, for example, you can drag and drop files from a local folder into the remote server. In the case of FTP it can be useful to have the ability to transfer groups of files and directories without typing, as it also allows a better interface with your working environment.

There are lots of resources on the Web for developers, some of which are freeware, shareware or proprietary. The resources you can find cover most things you will ever need; from fully featured Integrated Development Environments (IDEs), through to simple editors and graphic tools. What will you need to start developing applications? To some extent it depends on whether you take a minimalist approach or whether you prefer environments and tools that automate tasks for you.

You can make do with a simple text editor that probably already exists on your computer, your browser and an FTP client if you want to upload your files to a server. You can get editors that are specific to the languages you are using or download plug-ins to assist you. These exist for HTML, JavaScript, PHP, Perl and most other Web languages. There are syntax checkers, program coloring aids and even editors with built-in mini-servers to test code.

Development environments and editors will also automate tasks such as uploading pages via FTP or the batch conversion of files from one type to another.

All these various tools can be bought (as you usually would) from shops or online. There are also freeware or shareware versions where you pay a small fee or nothing at all. You may encounter the term open source while looking for suitable software, these are projects that are freely available to the general public and are usually ongoing, developed and supported by a volunteer community. The code behind such projects is usually available to view and alter. End users have the right to change such code and redistribute the software. Open source licenses may have some restrictions applied though, such as the requirement to preserve an author's name and copyright statement in the code.

Freeware can be distributed under a different kind of license where, although the code isn't available to view or alter, you do not pay a charge for it. It may possibly have a restriction such that it may not be sold on or used by government agencies or armed forces.

Shareware is usually software that is distributed ahead of payment, which is set to some point in the future so you have time to try it out.

Resources

- Open up your browser and, using a search engine, have a look around for various types of utility that may help you. You have various choices for types of license for software:
  - freeware is free to use and download
  - shareware to try and then pay later
  - off-the-shelf to buy.
- You may find FTP utilities and more secure software that use SFTP or FTPS.
- Freeware examples include: WinSCP, PuTTY …
- There are also lots of editors available for every kind of language and platform you are developing on.
- You may need tools later to convert files, such as graphics, between different formats.

1.7 The Web Browser

Possibly one of the main tools you will be using while developing Web applications is the Web browser. This connects, as we have seen, over the Internet to a Web server that answers requests from selections of hypertext documents. The HTTP protocol is used in this dialog. Pages are located through the use of URLs – Uniform Resource Locators which usually begin with `http://`, although most browsers will also support `ftp` and `https`.

Most browsers share common features in that they will display graphics and support many media features. However, this doesn't have to always be the case; there are browsers that allow the Web to be visited using text only, which is very fast.

The format of incoming Web pages is HTML which is interpreted by the browser and displayed as the instructions describe. In addition to HTML most browsers support other types of file such as JPEG, GIF and PNG. More can be supported using plug-ins units of software that can be added into the browser.

There have always been problems with standards and issues over compatibility, ever since browsers came about on various machines and platforms. When a new browser was made available there were things it did slightly differently, which led to that difference making a required change in HTML. In this way there was an evolution of the language over time, which allowed new browser features to be used, such as displaying images, changing colors or fonts. This led to several strands of the language developing and it therefore being non-standard. Standards were introduced to bring about the ability for pages to look the same no

CamScanner

matter what browser they were loaded into. There are now many modern browsers that will work within the standard versions of HTML and XHTML (Extensible HyperText Markup Language).

Browsers have expanded their capabilities beyond simple HTML rendering and often have support for IRC, newsgroups and email.

Choices

So, if you are going to develop applications, or even just Web pages, which browser do you choose? The best way to answer this is to ask yourself what you need and, probably more importantly, what your visitors are going to be using.

You could develop using a browser that you find you are comfortable with, then check on a few popular browsers. What features do browsers generally have? Obviously, their main function is to show text and graphics and possibly other media. Other features exist though that you should try and familiarize yourself with. For example, you should know that browsers can also open local files, usually under 'files' on the menu. A useful function of most browsers is the ability to look at a Web page's source code, which is usually under 'view' on the file menu. This can be useful when developing HTML, JavaScript or even to just check that the page you expect to load is the one displayed.

Most browsers include a section in 'preferences' or 'tools' where you can customize to some extent or set up special options. It's possible, for example, on most to set your home page (the place where the browser initially will go to, or visit when you click on the home button).

Browsers also use a cache, a local memory area, to speed up the loading of regularly visited pages. This is done by simply storing a copy in the client computer, which it will show when the page is requested again. You can usually adjust the size of this buffer, along with some other options such as how often a page in the cache is updated. Another option is to clear the cache, a very useful function at times! For example, if you are developing Web pages, what happens if a page you are working on gets cached? It may not update when you refresh (ask it to reload) the current page view. You could turn caching off or put a cache size of 0. As well as these methods there are ways of turning off caching through the Web pages themselves, which we will look at later.

Another set of options in your preferences on a browser relate to cookies. Cookies are small pieces of information that identify you to an application when you (re)visit the site. This may be to set up preferences so that when you do visit again, it knows what you like and will change the screen and greet you. This is not generally taken as foolproof for security; you will usually be asked for a password to verify if it is appropriate. For various reasons you may not want to be identified or have your habits tracked, so you can usually switch off the acceptance of cookies or set it so you're asked when one is presented.

Other options relate to the auto-filling of forms – do you want your details automatically added when a site gives you a form to fill out? Another useful option allows you to select where files that you want to download (such as MP3 or programs) will go.

Both cached pages and cookies can be cleared from memory, either individually or all together.

Concept Check

- Explore the various options of your favorite browser.
- How big is the cache?
- How do you clear all cookies and the cache?
- How do you stop cookies being accepted?
- Change the home page.
- Where do files go to that you download, the desktop or . . . ?
- Load a page from a Web site, or one of your own, and look at the source code through the browser – is it what you expect?

What is Inside a Browser?

On the client side probably the most important piece of software connected with the Web is the browser, which we will look at in more detail.

Figure 2 shows a conceptual diagram of the internals of a browser. As you can see it has various inputs and outputs with the outside world, or operating system in which it sits, the more important of which are outlined here. The controller acts as the main overseer of operations between the various components, for example it is responsible for collecting input from the user and also managing outgoing requests for the HTTP client to deal with for external communication. HTML and scripts are channelled through to the interpreters as needed and output from these is put through special display drivers. The browser can be extended with plug-in modules, which enhance capabilities such as multimedia, graphics and sound.

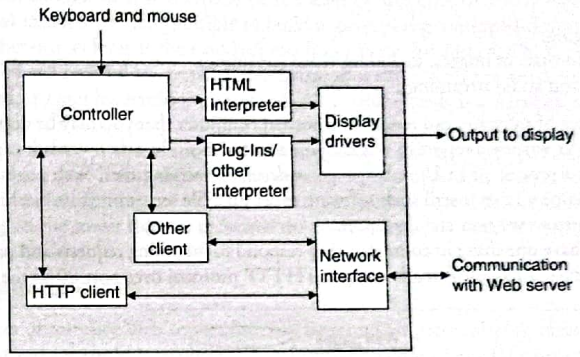


Figure 2 Architecture of a browser.

Concept Check

Using a search engine find which browsers are currently available.

What extensions and plug-ins are available for your favorite browser?

Checkpoint!

We now know:

- A browser contains quite a few interesting features beyond simple site visiting and viewing.
- A home page can be changed and set to whatever you want.
- A cookie is a small piece of information usually to identify a visitor the next time a site is visited.
- A cache is a memory area, or buffer, that is used to store Web pages that are frequently visited.
- How to change and view where downloaded files go
- A browser is a complex piece of software.
- It collects input from keyboard and mouse.
- It handles communication with Web servers.
- It interprets HTML and client side scripts.
- It has the capability to be functionally extended

CamScanner

1.8 What is a Server?

The word 'server' can have several slightly different meanings and contexts. Generally, in computing, it refers to a software application that provides a service on behalf of another piece of software called a client. It can also refer to the machine that runs the software or even in some instances the company that provides the service.

In the case of the software application, there can also be different kinds of server providing various services. For example, you can have a Web server but you can also have file servers and application servers. There are also mail and database servers. Here is a list of some of the possible server types:

1. Web.
2. File (possibly using FTP, for example).
3. Application.
4. Instant Messaging.
5. Image (many formats of images, including video streaming).
6. Sound (audio and audio streaming).

A server, as a piece of software, can reside on a normal computer that you may be doing some other task on at the time, such as writing a report or working on a spreadsheet. In other words it can be a background task (also known as a process or in Unix language, a *daemon*!), serving files, Web pages, audio streams or other multimedia. Some viruses install such software to set up a file server on your machine for illicit use of your computer resources over non-standard ports.

All these servers have one thing in common: they respond to incoming requests and provide a service. In the case of a Web server, a request is made using the HTTP protocol over port 80 to the server, which will

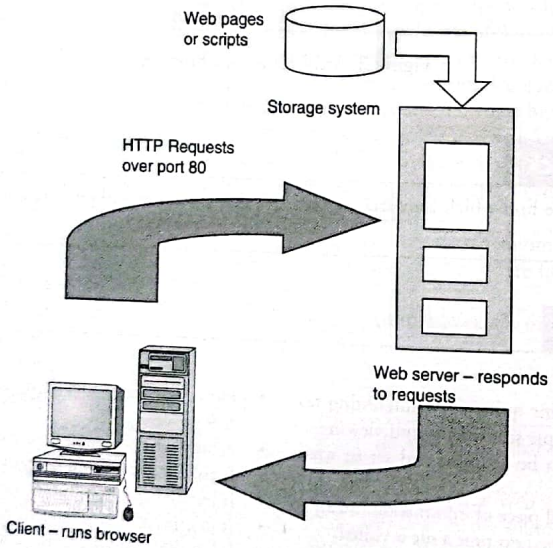


Figure 3 Client and (Web) server model.

then respond either with the requested item or an error message. A file server will respond, maybe through the FTP protocol, in a similar way. This model is known as client–server (as shown in Figure 3) where there is a client making requests and a server that deals with those requests. An example of another kind of model is peer-to-peer; here each node or computer is equal, in that they are both a server and client. This is popular for file sharing where a computer will respond to requests as to what files it has and, if asked to, will send a file to any members of that network. The real power lies in the effective distribution of computing power and bandwidth, rather than concentrating it in only a few servers.

A server can run on a fairly adequate computer that you would have at home or in the office but it can be a dedicated piece of hardware with high speed Internet links, terabytes of storage and massive memory. It can also have a multiple processor motherboard. This kind of server may have to deal with a great number of requests in quick succession and is typical of the kind of machine that may work at a Web hosting company or ISP for example. It's also possible to build a server using a stripped-down, maybe slightly out-of-date, machine because as long as the load isn't too heavy it can be quite a simple computer system. For example, it doesn't need heavy duty graphics processing so really it does not need a graphics card. It also doesn't need sound or complex hardware interfaces. All it really needs is a network connection, a decent amount of memory and disk space.

An interesting question at this point is: which do you think is more complex, the client or the Web server? Generally, the answer is that the server has a much more simple task, as all it has to do is respond to requests, whereas the client (in the case of a Web server) has to render the graphics, deal with input from the user, etc. Scripting on the server side has increased the complexity of its requirements though!

1.9 Server Choices

There are a number of software Web servers that can be set up on personal computers. Quite a few exist in the public domain and so cost little, if anything, to install. Here are a few of the many available:

1. Apache HTTPd Server from the Apache Project.
2. Internet Information Services (IIS) from Microsoft.
3. Personal Web Server from Microsoft (superseded by IIS).
4. Sun Java System Web Server (formerly Sun ONE Web Server) from Sun Microsystems.
5. Zeus Web Server from Zeus Technology.
6. Abyss Web Server from Aprelium Technologies.
7. AOL server from America Online, open source.
8. BEA WebLogic from BEA Systems.
9. Light tpd.

Some exist as part of a package that also includes scripting language module extensions (such as PHP) and a database technology (such as MySQL). These packages have been known under the acronyms LAMP, WAMP or XAMPP depending on which platform they are based on. For example, the Linux package is LAMP, standing for Linux, Apache (the Web server), MySQL and PHP (and/or Perl + Python) although even more variations exist.

There follows an overview of some of the most popular software.

Apache HTTPd Server

This is the most popular server on the Internet and is developed and maintained by an open community of developers under the Apache Software Foundation. Apache is highly configurable with many compiled modules available, ranging from server side programing support to authentication schemes. Perl (Practical Extraction and Report

Language), Python and PHP are supported, along with the authentication modules such as `mod_access`, `mod_auth`, and `mod_digest`. Other features include SSL and TLS support as `mod_ssl`, among many other options. Apache exists as two major versions: the older 1.x series and the more recent 2.x, which has several major enhancements including Unix threading and better support for non-Unix platforms such as Windows.

Available from: `http://httpd.apache.org/`

Internet Information Services

The main competitor to Apache is Microsoft's IIS or Internet Information Services (sometimes Server or System). This is a set of Internet based services for Windows machines that were originally part of Windows NT as an option pack and were then subsumed and integrated with Windows 2000 and Windows Server 2003. It includes servers for FTP, SMTP, NNTP and HTTP. Windows XP Professional includes a restricted version of IIS.

Available from: Microsoft (it could be built into your current Windows OS)

PHP, Perl, Python

There are quite a few server side languages available that can be downloaded and installed as you need them. In this book PHP is focused on as a popular, modern server side language, although we will look at some of the others to familiarize ourselves with important features.

PHP main site: `http://www.php.net/`
Perl main site: `http://www.perl.org/`
Python main site: `http://www.python.org/`

MySQL

One of the most popular relational database servers, utilizing SQL, on the Web is MySQL. It is multi-threaded and multi user. It is open source software that is available under the GNU General Public License (GPL) as well as other kinds of license.

MySQL main site: `http://www.mysql.com/`

Checkpoint!

We now know:

- A server is the software system that deals with incoming requests.
- It sometimes refers to the whole hardware computer (or even the ISP!)
- There are different kinds of serve: Web, file, application, Instant Messaging, multimedia (image, sound or video, sometimes streaming).
- A server can reside on the computing device you use for other tasks or be dedicated.
- There are lots of servers available for a variety of workloads, prices and platforms (the OS).
- Apache is one of the most popular and is available in special packages, complete with server side languages and database systems.
- There are other models besides client–server; these include peer-to-peer, where each node (a computer) acts as an equal client–server.

1.10 Setting up Servers

To get started developing on the server side you will need access to a Web server or to set up a server on your own computer. The server software can be downloaded and installed on your particular computer, the one you will be developing on. Alternatively, you may want to set up a computer on your LAN to be used

specifically for this task. If you do want to set up a full server you may want to consider the system from the 'ground up': What operating system is best? What hardware configuration would be most suitable?

The first point you should be aware of when finding a server is that some OS come complete with them built in; these include: Mac OS X, Windows (limited on some versions), Linux distributions (such as SUSE) and also Unix. However, even if they are installed they may need 'switching on' and configuring. In some cases it may actually be easier to download packages. This may also create a more powerful server.

Each of the individual packages, such as Apache HTTPd Server and PHP, is available to download and install. Usually the main Web site holds a binary (pre-compiled) copy for your platform (Windows, Mac, Unix, Linux) which you then download and follow the precise instructions, which vary with each operating system.

Another way to set up your machine is to use a package that includes everything you need. XAMPP is one of the easiest packages to install and it is described here.

XAMPP

One of the easiest server packages to set up is XAMPP (available at Apache Friends: `http://www.apachefriends.org/en/index.html`). This package is available for many platforms such as Linux, Windows, Solaris and Mac. It is very easy to install and free. While package contents do change it generally contains at least the following:

1. Apache HTTPd (the Web server).
2. MySQL.
3. FileZilla FTP Server.
4. phpMyAdmin.
5. OpenSSL (for security).
6. PHP.

First of all, download the XAMPP package from Apache Friends; you will also notice an XAMPP Lite package for smaller disk space use – the choice is yours. Extract the package to the root of a drive such as `C:\`.

After this unpacking you will find a directory named `C:\xampp` (or `C:\minixampp` for the Lite version).

If you want to run the servers as actual services under Windows NT/2000 then start the install batch file. Otherwise, you can manually start and stop the servers, which is fairly easy anyway. The way to start the Apache server is to simply look for and run the `apache_start.bat` file and to stop it simply close the console window.

To start the MySQL server, use `mysql_start.bat` and `mysql_stop.bat`.

Where is the Server?

Developing and experimenting with your own setup probably means you don't have to worry too much about your IP address and name resolution initially. You simply point at your own computer with the loop-back (so called because it points back at the machine itself!) address, `http://127.0.0.1` or `http://localhost`. If your Web server is a separate machine you need to know the local LAN address such as `http://192.168.1.8` and point at that. If you don't know the IP address, there are a few ways to find it out, depending on which platform you are on. Go to the server machine and, depending on the OS, use the appropriate command. Windows machines will respond with a list of current IP addresses assigned if the `ipconfig` command is given at the command prompt window. Most Unix-based and Unix-like machines (including Linux!) will give you a clue if you use `ifconfig`.

Figure 4 shows the output from `ifconfig` on a Mac OS X machine within a terminal window. This contains a list of all the devices connected to networks in the computer. In this particular output, `lo0` is the loopback `127.0.0.1` as you can see. `en0` is the normal Ethernet connection via cable (see the entry,

CamScanner

100baseTX <full-duplex>). en1 is the wireless connection. You could cross-check these entries with dmesg output on a Unix-type machine by looking at the MAC (Media Access Control, unique identifier) addresses (in this case, 00:10:93:72:f3:e2 for en0 and 00:12:24:2b:1f:af for en1).

Testing

Whatever package you choose the following testing should apply. To check that your server is running and actually serving pages you could try pointing your browser at http://127.0.0.1 or http://localhost. You should see a welcome screen rather than an error message of *page not found*!

```
Ralph-Moseleys-Computer:~ralphmoseley$ ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        inet6 ::1 prefixlen 128
        inet6 fe80::1 prefixlen 64 scopeid 0x1
        inet 127.0.0.1 netmask 0xff000000
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
stf0: flags=0<> mtu 1280
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST>
              mtu 1500
        inet6 fe80::20d:93ff:fe72:e3e2 prefixlen 64 scopeid 0x4
        inet 192.168.1.2 netmask 0xffffff00 broadcast
                      192.168.1.255
        ether 00:10:93:72:f3:e2
        media: autoselect (100baseTX <full-duplex>) status:
                      active
        supported media: none autoselect 10baseT/UTP
                      <half-duplex> 10baseT/UTP <full-duplex>
                      10baseT/UTP <full-duplex,hw-loopback>
                      100baseTX <half-duplex> 100baseTX
                      <full-duplex> 100baseTX
                      <full-duplex,hw-loopback>
en1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST>
              mtu 1500
        inet6 fe80::211:24ff:fe2b:fafprefixlen 64 scopeid 0x5
        inet 192.168.1.3 netmask 0xffffff00 broadcast
                      192.168.1.255
        ether 00:12:24:2b:1f:af
        media: autoselect status: active
        supported media: autoselect
fw0: flags=8822<BROADCAST,SMART,SIMPLEX,MULTICAST> mtu 2030
        lladdr 00:0d:93:ff:fe:72:e3:e2
        media: autoselect <full-duplex> status: inactive
        supported media: autoselect <full-duplex>
```

Figure 4 Network details using ifconfig.

The next question is: once you have your server running how do you publish your pages?

No matter what server you install, there will be a directory that acts as the place you put displayable Web pages. Table 5 shows the sub-directories under XAMPP. To get your own pages delivered by the server, locate the directory htdocs under your main directory for XAMPP, or under the Apache directory you have installed. For example it could be C:\xampp\htdocs or C:\minixampp\htdocs. Any pages you want to be served should be placed in this directory. So, if you had a Web page called myfirstpage.html, you would place it in the htdocs directory and then point your browser at:

```
http://localhost/myfirstpage.html
```

Hopefully, all being well, it should appear! Notice here that you do not mention xampp or htdocs in the browser address!? The reason is that htdocs is taken as the starting point for any Web materials. If you wanted one part of your Web site to be about your large collection of tropical fish and another to be about your interest in golf, you may like to keep the different materials separate so you would probably make a directory called fish and another called golf. Any Web pages could then be accessed by:

```
http://localhost/golf/
```

for golf materials and

```
http://localhost/fish/
```

for tropical fish. Don't forget that if someone writes in the address http://localhost/golf/, there must be a page with the name index.html or index.php in that directory in order for the start page to work.

Table 5 Directory for XAMPP

File or Directory	*Meaning*
\xampp\apache\logs	Where log files go for Apache and PHP
\xampp\apache\bin\php.ini	Central configuration file for PHP with MOD–PHP
\xampp\cgi-bin\	CGI-BIN directory
\xampp\apache\conf\httpd.conf	Central configuration file for Apache
\xampp\htdocs\	Web document root for HTML, PHP, CGI
\xampp\install\	Initializing documents
\xampp\mysql\	Directory for MySQL
\xampp\perl\	Perl directory
\xampp\php\	php.exe + dlls + pear
\xampp\phpmyadmin\config.inc.php	Configuration file for phpMyAdmin
\xampp\tmp	Temporary directory for PHP uploads and sessions etc.
\xampp\moddav	MOD-DAV example

CamScanner

External Access to the Web Server

To access the server from the outside world (beyond your LAN!) you need to know its external IP address. You can find this out from the computer or by using a Web service such as `http://www.whatis- myip.com`. Once the address is known you would then point a browser at this address plus the directory you want to see; for example: `http://86.135.74.177/golf`

An important point to mention here is that to access your pages you must configure any firewalls you have, either on your router/modem or your server machine itself, to allow access to port 80. Usually, firewalls have ways of letting the user configure so only the ports they want can be opened. Likewise, if you wanted to set up an FTP or other kind of server then you must open the appropriate port(s) for access. Don't forget also that some OS, such as Windows XP, have firewalls built in.

What if you wanted to associate a domain name with your server? To do this you need to purchase a domain name such as `http://www.mygolfempire.org`. This domain name is then registered through a DNS service such as `http://www.dyndns.org/`, which connects your IP with that particular name.

Checkpoint!

We now know:

- Server software can be downloaded and installed singularly or as special packages for various OS platforms such LAMP, WAMP or XAMPP
- A complete package may contain the server itself, server side languages and the database system, together with tools.
- A server you learn, or develop, on can be your usual work machine such as a laptop or a dedicated computer on your LAN.
- You can access a server on your own machine, usually through http://127.0.0.1, http://localhost or the LAN IP address.
- If you are using a dedicated server, use its LAN IP address. You can usually find this out through the network configuration system of that particular machine or through the command line: if config (for Unix-like computers) or ipconfig (for Windows machines).
- Most servers have a test page when you initially point the browser at its address such as http://127.0.0.1.
- There is a Web document root address for the server and individual users so pages can be published on the connected network.

1.11 Web 2.0

The Web 1.0 denotes the way Web is used in nineties i.e. from 1991 to 2000. The role of the user in Web 1.0 was merely as consumer of the content. That means user views the contents available on the different sites but not participating in a big way to create (or share) the content of the Web. The main focus of the Web 1.0 was on the publishing of the contents which users view and hence, it was one-to-one or one directional. Around 2000, the trend of using the Web started changing and focused on involvement of the users for creating, sharing, organizing the contents as well as collaborating with each other for collective intelligent. This new trend of using the Web is referred to as Web 2.0. The term Web 2.0 was first used by Darcy DiNucci in 1999 in her article "Fragmented Future":

"The Web we know now, which loads into a browser window in essentially static screenfuls, is only an embryo of the Web to come. The first glimmerings of Web 2.0 are beginning to appear, and we are just starting to see how that embryo might develop. The Web will be understood not as screenfuls of text and graphics but as a transport mechanism, the ether through which interactivity happens."

However, it has been made more popular by Tim O'Reilly and Dale Dougherty at the O'Reilly Media Web 2.0 Conference in late 2004.

Web 2.0 is not a standard in Web but it denotes trend which includes set of applications and technologies that focuses on user-generated contents, usability and interoperability (Refer Wikipedia). The shift is from just retrieval of information through Web pages to collaboration and sharing of the information. The shift is from traditional Web to community base Web where user gives their input, share the content, interact with each other and collaborate in many ways and sometimes real time control on the content. Considering all these facts, we can say the major elements of the Web 2.0 are:

- **User-generated content:** Users create, edit and share the contents.
- **Interaction:** The Web site promotes the interaction among the content creators, users and enterprises.
- **Social Participation:** Users connect socially with each other using social networking sites and shares photos and videos using Web 2.0 sites.
- **Real time control:** Users have control over the data in real time in many cases.

The examples of Web 2.0 includes social networking sites, blogs, photo or video sharing sites, Wikis, Social television sites, searching sites, tagging (folksonomy) and many more. Today, social networking sites are very popular which allow users to connect with each other and share the contents. The photo and video sharing sites allow user to upload their photos and videos and share them with friends and groups or community. The search sites like Google allow the user to search any content on the Web in an interactive manner using advance search features. The tagging allows user to classify the contents like photos, movies etc. We can describe the Web 2.0 also as:

- **Social Web:** Allowing end users to interact and encourage that end users are part of Web.
- **Reach Internet Applications (RIA):** Gives the new experience like desktop on Web both usability and graphical point of view.
- **Web Oriented Architecture:** Web 2.0 sites exposes their functionally to be used and integrated in other Web sites to make richer experience. One of the examples is web services which can be called and integrated in any Web sites. Another example includes mashup which uses contents from more than once source to create rich application.

The major technologies used to create the Web 2.0 sites divide into client side and server side technologies. The important client side technologies include AJAX (Asynchronous JavaScript and XML) and JavaScript. AJAX updates the selected part of a Web page without reloading it again giving high speed performance. For that AJAX uses JavaScript and Document Object Model (DOM). The AJAX uses XML or JSON (JavaScript Object Notation) formats for data representation for its request. The Adobe Flash also played an important role which now is supported internally in HTML5. The server side technologies include PHP, Python, Perl, Ruby, J2EE,. NET framework and many more. The machine readable data formats like XML, JSON, RSS plays important role as data available in these formats can easily be used by other Web sites.

CamScanner

Let us go through the examples of the Web 2.0 sites and applications in order to understand the significance of the Web 2.0.

- **Social networking sites:** It is one of the most popular Web 2.0 applications which allow user to interact online to increase the social network or business. It allows users to create their own communities of friends and professionals. The popular example sites are MySpace and Facebook. The Twitter, LinkedIn, Google+ are also examples of social networking.
- **Wikis:** The Web sites that allows users to edit the contents and contribute. The most popular example is Wikipedia.
- **Blogs:** Web sites that list the entries in reverse chronological order. It allows the user to participate by writing their own blogs and become a major social phenomenon. The examples of blogs are Blogger, WordPress.com etc.
- **Searching:** There are many online searching sites allowing user to enter a word or phrase and list the pages or sites containing that word or phrase. The very popular search site is Google. There are also other sites like Yahoo, MSN etc. There are also sites available for vertical search i.e. allowing search in a specific area like a travel. The location based search sites are also available.
- **Social Media:** The social media sites allow the user to share the media like photos, videos, music, news etc. online. The YouTube is the most popular video site which allows user to upload the videos. It also allows user to rate or comment the videos uploaded. Joost is an example of the social television. Podcasting is also an example of social media which allows user to create their own audio or video. For example, company creates video to demonstrate their product. The Dig and Instagram also falls in this category.
- **Social bookmarking:** The Web sites that allow you to share Internet bookmarks, i.e. your favourite sites, blogs, etc.
- **User-generated contents:** Many Web 2.0 sites rely on the contents generated by their users. The online auction site eBay, online job search engine Monster and Amazon are such examples. The collective intelligence which enables smart ideas based on collaboration of user community in specific area also falls in this category. This has also given rise to the concepts like crowdsourcing, crowdfunding etc.
- **Mashups:** Mashup uses the functionalities of two or more sites or applications to create a new site or application which gives more value to the user. For example, combining Google map with local data in Web site adds more value.
- **Widgets:** The Widgets denote a small code that executes in HTML page automatically. Many social networking and blogging sites uses widgets from the Web to provide the different contents like announcements, press release or functionalities like clock, weather etc. The Google Gadgets and Yahoo Widgets are the examples.
- **Folksonomy and Social Tagging:** The tagging means labelling the contents for easy identification. The folksonomy refers to classification based on tags.
- **Really Simple Syndication (RSS):** The RSS is a mechanism that allows the distribution of news and updates from the Websites that updates frequently. The RSS is an XML format includes news and updates which are sent to user computers automatically over the Internet. The user installs RSS aggregator software and subscribes to particular channel to get the updates from the Web site as when available.

The term Web 3.0 is already coined and has started making sense which represents the intelligent Web which refers to the Web which understands the user needs and satisfies it. It is also popularly referred as Semantic Web. It is considered as future Web as it emphasize on meaningful Web.

Checkpoint!

We now know:

- Web 1.0 refers to the existing passive Web where user simply views the content.
- Web 2.0 refers to the trend in Web after year 2000 which emphasize on user-generated contents, interaction, social participation and real time user control.
- The Web 2.0 sites use AJAX and JavaScript on client side whereas PHP, Perl, Python, Ruby, J2EE, .NET etc. as server side technology.
- The Web 2.0 applications include Social networking, Wikis, searching, Social media, Social bookmarking, Collective Intelligence, Mashups, Widgets, tagging, RSS and many more.
- Semantic web is future Web also called Web 3.0 refers to more intelligent Web.

Summary

1. The Internet was developed as the need to communicate information between remote locations became evident.
2. The WWW is a large subset of the Internet and was developed as a way of connecting information that could be universally accessed.
3. The main model for communication on the WWW is client-server based.
4. HTTP was developed as a protocol to work for document transfer and is essentially request/response in nature.
5. Other protocols exist for different uses, such as the transfer of files between machines.
6. Many useful command line and applications exist that utilize the various protocols to help with file transfer and remote access.
7. Some problems do exist with certain protocols, which can be overcome.
8. Security should always be considered when transferring or accessing information over the Internet.
9. A browser is a complex piece of software made of several components including network, display and script interpreter modules that can further be extended by special multimedia plug-ins.
10. A Web server is a software that is active and responds to HTTP requests for Web pages.
11. Web 2.0 is a modern web that gives new kind of experience to the users with features like Interaction, Real-time user control, Social participation and sharing of information, collaboration and User generated contents.

CamScanner

Key Words and Phrases

Apache HTTPd A popular open source Web server for many platforms.

Cleint-server A model for network application architecture in which there are defined clients and separate servers.

FTP File Transfer Protocol, a standard for transferring files between computers.

FTPS File Transfer Protocol over SSL, another version of FTP running over SSL/TLS for security.

GET HTTP Method that passes parameters in Query String.

HTML HyperText Markup Language.

HTTP HyperText Transfer protocol, used to transfer HTML pages.

HTTP request Format used by HTTP to make a request for a Web page from Web browser.

HTTP response Format used by HTTP to send response from Web server.

Internet When written with a capital 'I', this refers to the publicly available system of interconnected networks that communicate standardized protocols such as IP.

Interpreter A software component that analyzes a script and acts on the commands given.

IP Internet Protocol, one of the main protocols used on the Internet.

Peer-to-peer A model for network application architecture in which each node can equally function as client or server.

Plug-ins Additional software components that extends a browser's capability.

Port An interface for communicating with a computer program over a network.

POST HTTP method that passes parameters in message body.

Protocol A standard, or set of rules and conventions that enables communication between two systems. A protocol can exist in hardware terms as well as software.

RFC Request For Comments, documents describing Internet protocols.

SFTP Secure Shell (SSH) File Transfer Protocol, a more secure protocol for transfer of files.

SMTP Simple Mail Transfer Protocol, the standard text-based method for transferring email.

SSH Secure Shell, both a protocol and a command line program for connecting to remote computers.

SSL Secure Sockets Layer, a secure protocol with encryption.

Status code Code used by HTTP to inform client about the request.

TCP Transmission Control Protocol, a connection-oriented protocol with reliability, working at the transport layer level.

TLS Transport Layer Security, the successor to SSL with few differences between SSL 3.0 and TLS 1.0.

UDP User Datagram Protocol, a fairly minimal message-oriented transport layer protocol, providing no guarantee for message delivery.

URI Uniform Resource Indicator, an Internet protocol element that consists of a string of characters that indicate a name or address referring to a resource.

URL Uniform Resource Locator, a standardized address for some resource on the Internet or elsewhere; it is a type of URI.

Web 1.0 A term used to refer the static Web.

Web 2.0 A term used to refer the modern web supporting user generated contents, collaboration and sharing of information.

WWW Refers to the World Wide Web, an information space and large subset of the Internet, where items (known as resources) are accessible via links.

Multiple-Choice Questions

1. Each resource in Internet is identified by an
   (a) MAC address
   (b) IP address
   (c) Device address
   (d) None of the above
2. DNS stands for
   (a) Device Number System
   (b) Domain Number System
   (c) Domain Name System
   (d) None of the above
3. Which of the two following layers are at the heart of the Internet?
   (a) Data Link and Transport
   (b) Transport and Application
   (c) Application and Network
   (d) Transport and Network
4. Which of the following protocol deals with the file transfer?
   (a) FTP
   (b) Telnet
   (c) HTTP
   (d) All of the above
5. Which of the following is the most important protocol for WWW?
   (a) FTP
   (b) SMTP
   (c) HTTP
   (d) All of the above
6. Which of the following is not an Email-related protocol?
   (a) SMTP
   (b) POP3
   (c) TCP
   (d) All are email protocol
7. Which of the following HTTP error code indicates page not found?
   (a) 404
   (b) 200
   (c) 500
   (d) None of the above
8. Which of the following HTTP method uses message body to transfer parameters?
   (a) GET
   (b) POST
   (c) Both of the above
   (d) None of the above
9. Message body of the HTTP response consists of
   (a) Implicit headers
   (b) HTML document
   (c) Both of the above
   (d) None of the above
10. Web browser works as
    (a) Web client
    (b) Web server
    (c) Both
    (d) None
11. To request for a page on WWW in Internet, you need to specify
    (a) Page address
    (b) File name of a page
    (c) URL
    (d) Any of the above
12. URL stands for
    (a) Uniform Resource Location
    (b) Uniform Resource Locator
    (c) Uniform Reserve Location
    (d) None of the above
13. How can we enhance the capabilities of the browser?
    (a) By adding plug-in modules
    (b) By installing new version
    (c) By changing browser vendor
    (d) None of the above
14. The term 'Server' in context of software means
    (a) a software running centrally
    (b) a piece of software providing services
    (c) a software running on dedicated hardware
    (d) All of the above

CamScanner

15. Which of the following is a Microsoft Web server?
    (a) Apache HTTPd
    (b) Internet Information Services
    (c) Sun Java System Web Server
    (d) None of the above
16. Which of the following is not part of XAMPP server?
    (a) MySQL
    (b) PHP
    (c) Apache HTTPd server
    (d) All are part of XAMPP server
17. Web document root for HTML, PHP, CGI etc. in XAMPP is
    (a) \xampp\php\
    (b) \xampp\htdocs\
    (c) \xampp\perl\
    (d) None of the above
18. Which of the following indicates local access to the Web server
    (a) http://127.0.0.1
    (b) http://localhost
    (c) Any of the above
    (d) None of the above

Review Questions

1. What is an OSI model? List its layers along with name of example protocols for each layer.
2. Give the Internet layered model with protocols for each layer and compare them with OSI layers.
3. Give the function of each layer in brief for the Internet layered model.
4. What are the important aspects to be considered for communication between machines on Internet? Name few protocols or programs used for communication on Internet.
5. What is FTP? What is its use? List the commonly used FTP commands.
6. List the problems associated with use of ftp for file transfer.
7. List the three major applications used on Internet by people for day to day activities.
8. What is Telnet? How can you use it? List its limitations. How does it differ from SSH?
9. How do Web browser and Web server communicate to transfer a Web page? Give each step with proper example.
10. Explain the HTTP GET request and corresponding response with the help of an example.
11. List the ports used by the FTP, Telnet and HTTP.
12. What is HTTP? What is its use? List the major components of HTTP and explain them in brief.
13. Give the format of HTTP request and response. Explain both giving an example.
14. What is use of message headers in HTTP request and response? Discuss with examples.
15. Differentiate between the GET and POST methods.
16. How do you make your Internet connection secure? Name the protocols and programs used for secure connection for various purposes.
17. What is Web browser? List the various capabilities normally included in almost all the Web browsers.
18. What is advantage of caching Web pages? What is its disadvantage? How can you stop browser to stop caching the Web pages?
19. Draw the architecture of a browser and give the function of each component.
20. What is role of software server? Name the different servers based on the services they provide.
21. What are the major resource requirements for running a software server?

22. What is role of Web server? List the popular Web servers.
23. Write a short not on Web servers.
24. Give the features of Apache HTTPd and IIS servers.
25. What is XAMPP server package? What servers does it contain? How would you test the working of XAMPP once installed?
26. How do you access any Web server internally from same machine and externally from other machines?
27. What is Web 2.0? How does it differ from Web 1.0. Give its characteristics.
28. List and explain the Web 2.0 applications.

Practice

1. Use online resources to configure your browser to provide secured and efficient Internet access.
2. Learn HTTP protocol in detail using latest version of RFC describing the HTTP protocol.
3. Download Apache HTTPd server, PHP and MySQL as separate pack, install them and configure them to work together.
4. Download XAMPP server, install and test its working.

Chapter Quiz

1. See if you can find out a little more about the origins of the Internet by using search engines such as Google. You can also use the useful Web addresses at the end of this chapter.
2. Other models of communication exist than just client–server as in the case of using a browser. Can you think of any software you use that may use a different kind of approach?
3. Is the computer you use for development protected by a firewall and, if so, how is it configured to allow various communications through the ports, such as FTP?
4. Survey about the most popular Web servers and find their market shares.
5. Explore the meaning and use of the semantic web with its standards.

Useful Web Addresses

http://www.w3.org/
http://www.isoc.org/internet/history/
http://en.wikipedia.org/wiki/Internet
http://www.w3schools.com/
http://httpd.apache.org/
http://www.apachefriends.org/
https://en.wikipedia.org/wiki/Web_2.0

CamScanner

Answers

1. (b)
2. (c)
3. (d)
4. (a)
5. (c)
6. (c)
7. (a)
8. (b)
9. (b)
10. (a)
11. (c)
12. (b)
13. (a)
14. (b)
15. (b)
16. (d)
17. (b)
18. (c)

2 Designing an Effective Web Site

LEARNING OBJECTIVES

At the end of this chapter, you will be able to:

- Define and understand the terms Web page, Web site and WWW.
- State the various Web site design issues.
- Understand the effect of browser compatibility, bandwidth and cache, and display resolution in Web page design.
- Define the 'Look and Feel' of a Web site and discribe the various elements of look and feel.
- Describe the methods for designing page layouts and linking them for effective navigation.
- Identify the location of the information on screen, based on importance.
- Make your Web page design that is user-centric.
- Define the sitemap.
- List the steps to plan a Web site.
- Understand navigation of Web site and the important tips to design effective navigation.

The aim of this chapter is to explore the basic Web site design issues for making Web site effective and useful. You will learn about basic design principles, how to plan the Web site, organization of contents and navigation design of the Web site.

Designing an effective Web site needs to consider many issues such as browser support, bandwidth of the connection and cache support, display resolution, look and feel, Web site theme, page layouts and linking, user interaction, locating information and providing the sitemap.

Web site development is a project, which needs careful planning in order to be successful. The planning of Web site includes deciding the objective and goal behind preparing the Web site, knowing about audience for whom the Web site is designed, identifying and organizing the contents using different structures, preparing the prototype of the Web site and taking necessary steps to implement and publish the Web site.

Web pages in a Web site are linked together by hyperlinks. Users navigate through the contents using these links. Navigation is most important and crucial issue in good Web site design. It decides the way user can move from page to page and follow the different paths.

2.1 Introduction

Internet and WWW has made a revolutionary change in every field of life. WWW has influenced all the activities of human being. WWW is a collection of huge amount of Web pages grouped in the form of Web sites and hosted on Web servers. Web site is a collection of Web pages linked together to fulfill specified purpose and goal. Depending on the purpose and goal, Web sites are designed for personal information,

CamScanner

business organizations, education institutes, government departments, online marketing and selling, banking and insurance, entertainment, online booking and many more.

Designing a Web site that is effective and useful is a challenge as a number of factors influence this. A Web site operates in environment of software such as browser and operating systems. Pages are displayed on different display devices. Users have diverse expectations from the Web site. Decisions made by Web site designer are crucial. All of these factors are very important in designing a successful Web site in terms of achieving the goals set for the Web site. Considering all these common factors, a Web site is planned. Planning includes objective and goal, audience characteristics, contents, prototyping and taking necessary steps to implement and publish the Web site. Good navigation makes a user comfortable in easy access and deciding viewing paths.

2.2 Web Site Design Issues

Let us consider the common issues that are applicable while designing an effective Web site that is portable on different browsers, attractive in terms of colors and graphics, rich in contents, easy to read and navigate and downloads faster.

Browser and Operating Systems

Web pages are written using different HTML tags and viewed in browser window. The different browsers and their versions greatly affect the way a page is rendered, as different browsers sometimes interpret same HTML tag in a different way. Different versions of HTML also support different sets of tags. The support for different tags also varies across the different browsers and their versions. The Web technology is also moving from HTML to XHTML. Older version of a browser may not support recent tags and features. Many users still follow the older versions of browsers although newer versions are available. For example, if your page is designed with Cascading Style Sheet (CSS) and if the user browser does not support CSS, the page looks very different. Same browser may work slightly different on different operating system and hardware platform. Hence, the browser compatibility is a critical issue to make your pages portable on the different browsers with different versions.

To make a Web page portable, test it on different browsers on different operating systems. Use the development tools to add special features to your pages supported by majority browsers. The best way to avoid these problems is to follow the standards. Use the validation rules of the W3C to validate your code. Code that complies to the W3C standard improves the browser compatibility of your code as latest versions of popular browsers such as Internet Explorer, Firefox, Netscape etc. have started supporting the standards.

Bandwidth and Cache

Users have different connection speed, i.e. bandwidth, to access the Web sites. Connection speed plays an important role in designing Web pages. If user has low bandwidth connection and a Web page contains too many images, it takes more time to download. Generally, users have no patience to wait for longer time than 10–15 seconds and move to other site without looking at contents of your Web page. Hence, while designing Web pages, you have to also consider the low speed users like dial-up connections (having 56–256 Kbps) and hence limit the graphics, i.e. images in a single page to download it faster. However, high-speed connection with broadband technology has solved this problem significantly. With high-speed connections, more attractive graphics can be used to design visually attractive pages.

Browsers provide temporary memory called *cache* to store the graphics. When user gives the URL of the Web page for the first time, HTML file together with all the graphics files referred in a page is downloaded and displayed. At the same time, the images downloaded are also stored in cache area of

browser for a specified time as per user's preference. When next time the same page is visited, the browser gets only HTML file and uses the images from the cache which increases the download speed significantly. This feature is very helpful specifically on low-speed connections.

Display Resolution

Display resolution is another important factor affecting the Web page design, as we do not have any control on display resolution of the monitors on which user views our pages. Display or screen resolution is measured in terms of pixels and common resolutions are 800 × 600 and 1024 × 786. With the reduction in cost of large size monitors, 1240 × 1024 resolutions will become common in near future. Today, most monitors are using resolution of 1024 × 768 pixels. However, many old monitors are working on 800 × 600. Considering the different resolutions, we have three choices for Web page design.

1. Design a Web page with fixed resolution say 800 × 600. The Web page properly fits into a screen with this resolution but leaves some part of the screen if resolution is 1024 × 768. On the other side, if design is fixed to use 1024 × 768 pixels, it will not fit into the screen having 800 × 600 resolution.
2. Make a flexible design using HTML table to fit into different resolutions. Web page is divided into three columns with middle column having a variable width. Depending on the resolution of monitor on which page it is viewed, it changes its width. However, challenge is to design that part in such a way that on different screen resolutions, the content remains in proper viewable form.
3. If the page is displayed on a monitor with a higher resolution, the page is displayed on left-hand side and some part on the right-hand side remains blank. We can use centered design to display page, e.g. in middle of the screen, leaving equal space on both the side of the page. Figure 1 shows the home page of www.w3schools.com having centered design.

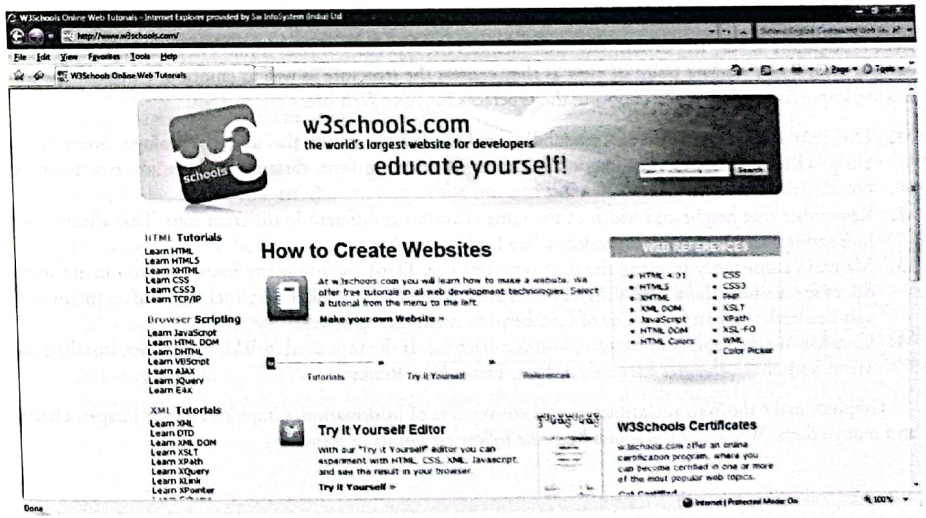


Figure 1 Home page of w3 schools.

We should also consider a point that a when page is seen on print media and screen media, it differs in colors due to different nature of both. While designing a page we have to keep in mind that it is to be displayed on screen media and accordingly the colors should be chosen.

Look and Feel

Look and feel of the Web site decides the overall appearance of the Web site. It includes all the design aspects such as Web site theme, Web typography, graphics, visual structure, navigation, etc. We will discuss some of them in this section and others in subsequent sections.

Web Site Theme

Web site theme emphasizes on the unification of design that should be reflected in each element of the design such that all pages of the Web site hold together and give the impression of a single unit. Web site theme gives the impression to the users that they are on same Web site while navigating through various pages. Every organization has unique reason to create their Web site. Web site theme should reflect the objective of the organization and convey the message of the organization. Following are some examples of the way we can choose the theme for a Web site:

1. Use logo of the company as theme. In this case, it must appear on each page to convey your message. For example, Telecommunication Company can use well-design logo and pictures throughout the site.
2. Use color scheme as theme. In such case, all your buttons, links, titles, labels and other text should use it consistently. Observe the consistency of color scheme throughout Web site in a popular Web site for learning Web technology (visit www.w3schoos.com).
3. Web site for global warming can use pictures, message related to cause and effects of the global warming on lives.

Fonts, Graphics and Colors

Web typography is very important issue that includes the typefaces of your fonts. They are important from both scanning and reading point of view as they express the structure as well as emotion. Consider the following important points while choosing the typefaces for your Web site.

1. Different fonts have different readability and it directly affects the user psychology. Some fonts are good for continuous reading while others for viewing from distance such as advertisement on hoardings.
2. Remember that height and width of the same character is different in different font. This affects your line endings and paragraph boundaries (see Fig. 3).
3. Maintain consistency in using the font type and size. Don't use too many fonts with too many sizes. Select few and use them with different sizes and modes (bold, italic, underline) for headers (main and sub-headers), titles and text. Use of CSS helps in maintaining consistency.
4. Consider the availability of fonts on visitor's machine. If the font used in Web page is not installed on visitor's machine, browser uses default font, Times New Roman.

Graphics make the Web site attractive and convey lots of information. Graphics include images, charts and many others. While using graphics keep the following points in mind:

1. One of the important points is the graphics file format. Different file formats support different levels of compression and other features. Popular image formats on Web are JPG, GIF, PNG, etc. Use them at right place considering their features.
2. Too many graphics or images with larger sizes reduce the download speed. Use limited graphics or use images with smaller size or in compressed format.
3. Use images that suit the theme of the Web site and profile of target audience rather using any image just to include graphics.

Colors are part of everybody's life and Web sites are not exceptions. People's liking are different for colors. Web site with good content but improper color scheme does not attract viewers and hence it fails. Hence, color scheme that suits your theme, content and liking of your target audiences plays a critical role.

For the graphic and color schemes, help of graphic and art designers is required to give the Web site commercial and professional touch.

Presentations and Access

Web sites are designed to provide contents online to the diverse group of visitors having different reading habits and expectations. A clear presentation helps in accessing the desired information quickly and easily without any hurdles. The visual structure of Web page, navigation links and use of the white spaces decides your presentation clarity. Consider the following points:

1. Web page should be divided in visually different areas to present the contents with different importance. Use grid-based structure to divide the Web page into different rows and columns. We will discuss page layout in detail in next section.
2. Make different parts of the page area visible by using the white space, i.e. the blank areas between them to make the contents eye-catching.
3. Keep the page simple and focused if required. For example, home page of http://www.google.co.in (see Fig. 2) uses very simple design that is focused to quick search, as it provides text field in middle area leaving other area as white space.
4. Do not overload a page with too much content. Divide the contents into parts, i.e. sections and within section pages and link them using hyperlinks.
5. Don't make page too lengthy. Instead, divide it into multiple pages with links such as "previous" and "next" to move forward and backward. For example, when google search results into many links, only a few are shown, and at the bottom links to other pages are provided with page numbers.
6. If a long page is to be designed, provide the link to top of the page at the bottom to quickly return to start.
7. Links between the pages and sections should exhibit consistency in appearance and meaning. Otherwise, that makes navigation difficult for the visitors.
8. Navigation, i.e. transitions between pages, should reflect the consistency in terms of graphics for links and their placement. Put the common links at the same place on each page. For example, link to home page is placed to top corner on each page. More about navigation is discussed later.

CamScanner

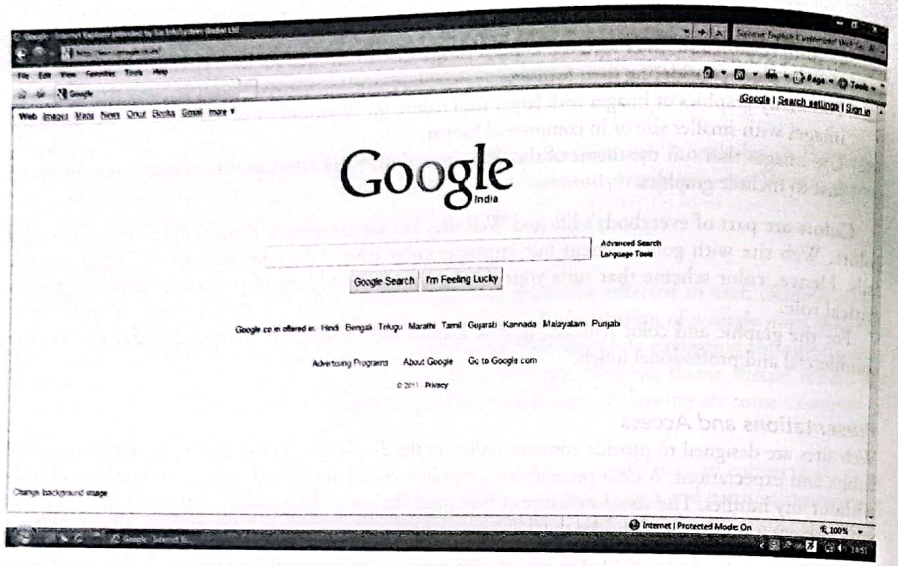


Figure 2 Example of simple design.

Concept Check

- Visit the Web site http://www.w3schools.com to study the different aspects discussed in this section.
- Find some example Web sites from WWW that strongly reflect their Web site theme in terms of their business objectives.
- List Web sites with attractive presentation. Try to find and write the reason why you feel that they are attractive.

Page Layout and Linking

Web site consists of individual Web pages that are linked together using various navigational links. Page layout defines the visual structure of the page and divides the page area into different parts to present the information of varying importance. Page layout allows the designer to distribute the contents in a page such that visitor can view it easily and find necessary details.

We can use HTML table element to design the grid-based page layouts. Figure 3 shows one possible layout designed using table element. We can design desired layouts using table element and their attributes with appropriate use. Complex layouts can be designed with nested tables. Use of table element for designing page layout is a usual choice as all the browsers support it. Alternatively, we can use CSS to design the page layouts. With advance CSS features available, it is possible to design complex and attractive layouts. However, in such a case browser support is an issue, as older browser may not support the CSS.

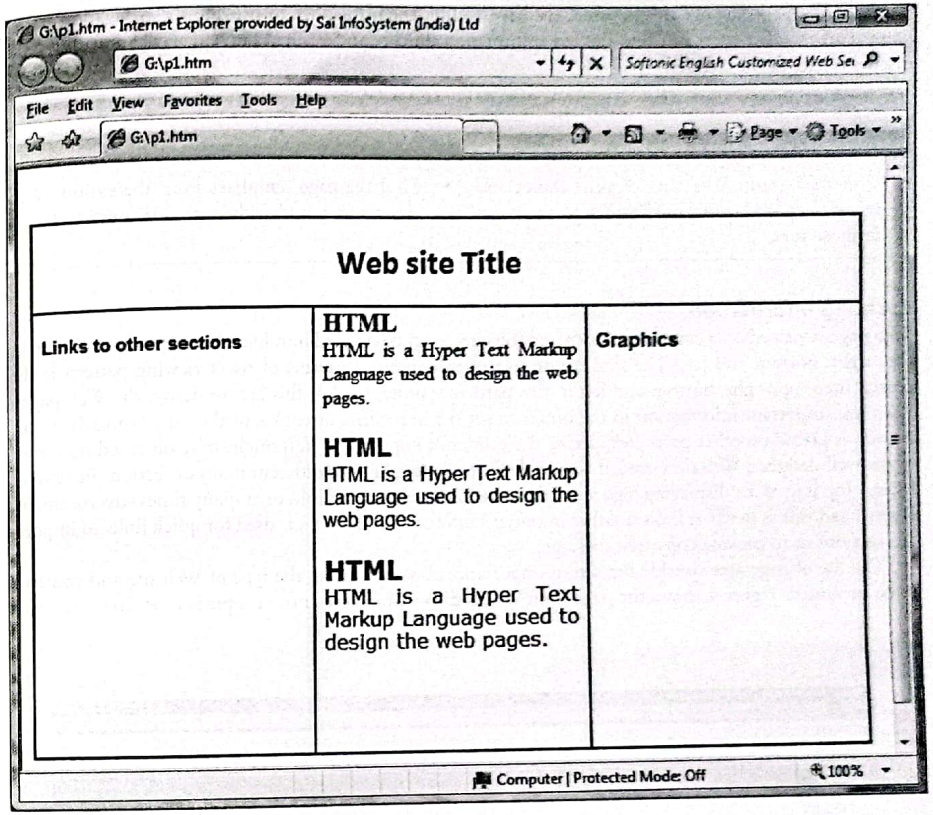


Figure 3 Page layout designed using HTML table.

Page layout is also called the page template. Many readymade templates are available on the Web sites which can be directly used to design the pages quickly by simply adding the contents into it.

Keep the following points in mind while designing layouts:

1. While distributing the contents in layout, balance between text and graphics. Too much graphics may lead to slow download of the page.
2. Locate the items on the page using screen importance to attract the user's attention on important items.
3. Maintain consistency among the layout of the pages such that it creates cohesive design of Web site matching to your theme. For example, www.w3schools.com uses layout consistently throughout the Web site.

CamScanner

Once the page layouts are prepared, we have to link them together to provide the navigation paths consistently so that users can find desired information quickly and easily. Different navigation styles are discussed later in the chapter.

Concept Check

- Visit the various Web sites of your choice and study the page layouts used to design the pages in those sites.
- Find the page templates from the various Web sources and modify them to suit your needs.

Locating Information

Web page is viewed on a computer screen and the screen can be divided into five major areas such as center, top, right, bottom and left. The first major area of importance in terms of users viewing pattern is the center, then top, right, bottom and left in this particular order. Exploit this fact to design the Web pages. Put most important information in the center to get the immediate attention of the user. Similarly, information is placed on other parts considering their order of importance. You might have observed that most of the well-designed Web sites use left-hand side for providing links to different items or sections in form of menu. Top is used for displaying logo and title as well as flashing news links or at many times advertisement. Right-hand side is used for links to other important information. Bottom is used for quick links to important sections or to provide copyright message.

The use of page area considering screen importance also depends on the type of Web site and contents to be presented. Figure 4 shows the page from Wikipedia in which the current topic is given in center with

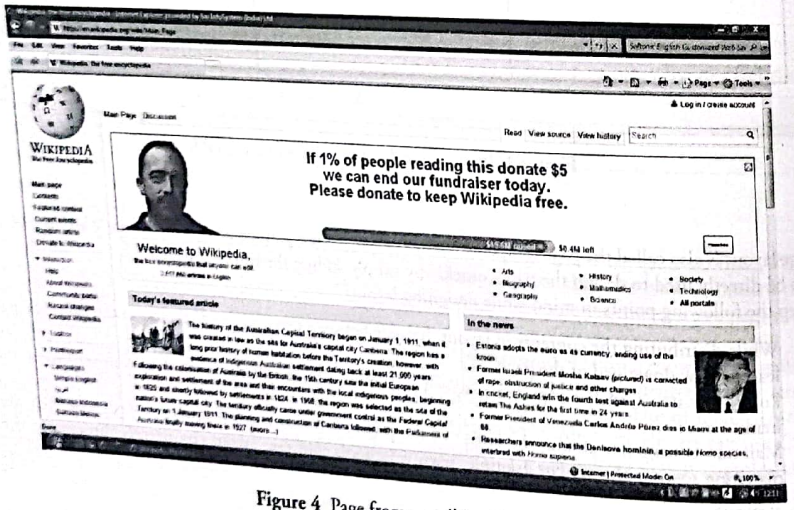


Figure 4 Page from en.wikipedia.org.

links to the news on right-hand side, advertisement on the top and links to other sections on left-hand side. Observe that search is provided on top right-hand side. Observe a page from Yahoo India to identify the different screen areas used with information in order of importance.

Make Design User-Centric

It is very difficult for any Web designer to predict the exact behavior of the Web site users. However, idea of general behavior of common user helps in making design of the Web site user-centric. Users either scan the information on the Web page to find the section of their interest or read the information to get the details. If the information is provided for reading, e.g. article on an online magazine or newspaper site, then user's general reading habit gives the idea of how to organize such information. People normally read from left to right and top to bottom. Hence, we can organize such contents in column fashion on a Web page. See Fig. 4 where article is provided as column in center. On the other hand, people scan TV screen in circular fashion in clockwise direction. This property can be used to provide the links to different sections of Web site on a home page.

For articles or tutorials divided into sections, providing the table of contents greatly helps users in quickly accessing the desired part of tutorial. The links in table of contents change colors once visited, so the user gets an idea of which links have been visited and which are yet to be visited. Figure 5 shows the article on computer software on the Wikipedia with table of contents (see en.wikipedia.org).

Web pages are linked together to provide non-sequential but quick reading. It is sometimes necessary to highlight the words in a paragraph as links to provide the links to the pages giving detail meaning of the word. Wikipedia uses this feature nicely to explore the meaning of important words in paragraph. See Fig. 5 describing computer software. Each important word is given in blue color and it is linked to the description

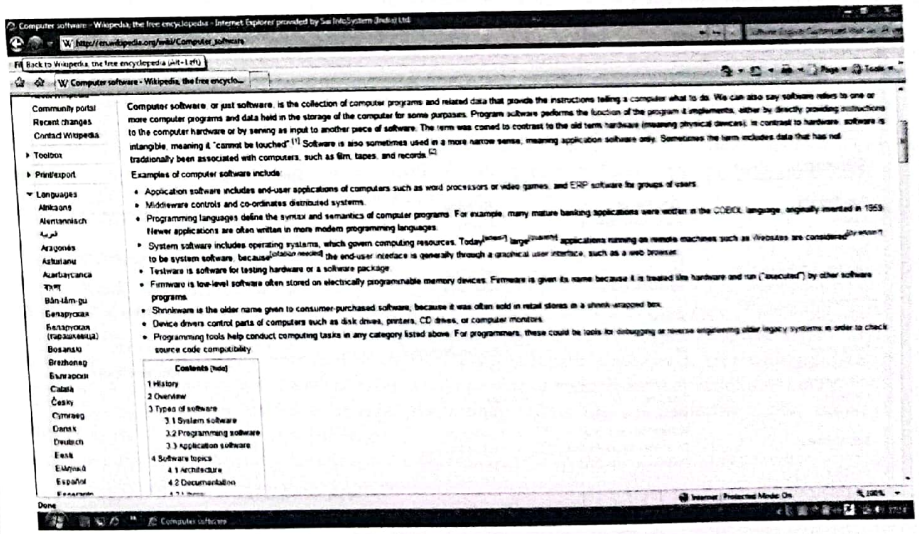


Figure 5 Page with table of contents.

CamScanner

of that word in detail, leading to a page which further uses this concept. User in this way can explore the given item upto the desired depth.

Some of the Web sites also provide links to the related topics at the end of the Web page describing a topic. This provides user with facility to read the related information to get more clarity.

Sitemap is also part of user-centric design for quick access to the information. It is discussed in next section.

Sitemap

Many a times Web sites are too complex as there are a large number of sections and each section contains many pages. It becomes difficult for a visitor to quickly move from one part to another. Once the user selects a particular section and pages in that section, the user gets confused about where he/she is and where to go from there. To make it simple, keep your hierarchy of information to few levels or provide the navigation bar on each page to jump directly to a particular section.

Another solution is to provide the sitemap including links to each section and their pages directly. Sitemap gives the organization of contents of a Web site graphically. Figure 6 shows the sitemap of the Yahoo News. By clicking any link on sitemap, you can directly jump to that page. Provide a link to sitemap with a button 'Sitemap' on each page of your site at same location. At any time user can use 'Sitemap' link from the current page and using the links on it, the user can quickly select the desired content.

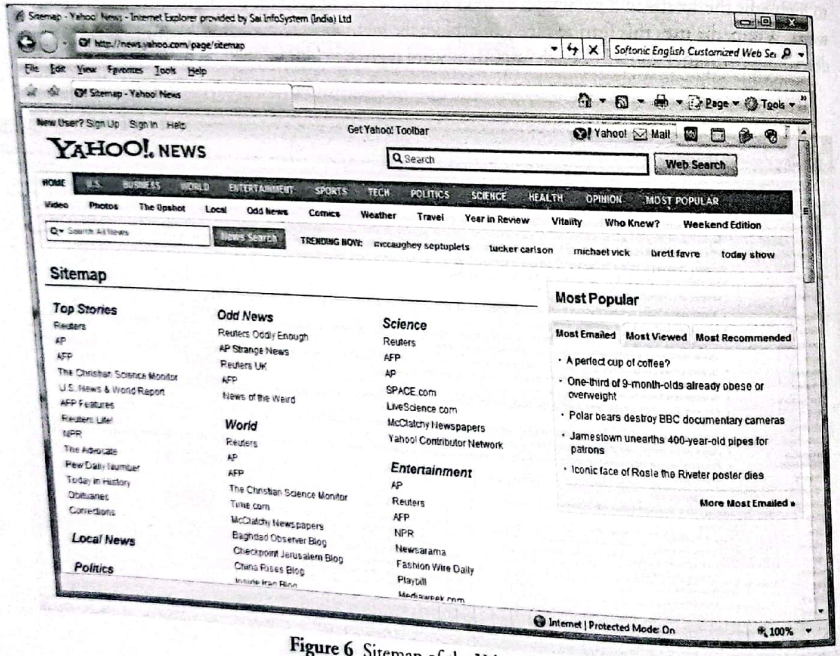


Figure 6 Sitemap of the Yahoo News.

Checkpoint!

We now know that:

- Browser compatibility is an important issue in a Web design. Web pages should be designed to make them portable on different browsers.
- Low-speed connection and cache memory support from browser are kept in mind while designing pages with graphics.
- Display resolution forces us to design page with fixed, flexible or centered design.
- Look and feel of the Web site includes all the design elements of the Web site.
- Page layout defines the visual structure of the page and links are part of navigation.
- Contents are placed on Web page considering the screen importance. Most important information is placed in the center.
- User-centric design provides the convenience to the user for getting desired information quickly and easily.
- Sitemap contains links to all the sections and pages to quickly move to the desired page.

2.3 Planning a Web Site

The most important activity in a Web site development is planning. To achieve higher success of the Web site in terms of user satisfaction, better planning is needed. Before we start developing a Web site, we should ask questions such as

1. Why are we developing this Web site?
2. What do we achieve by developing this Web site?
3. Who are the people who will use this Web site?
4. What are the information contents?
5. How are these contents organized? What are the possible ways?
6. How the files prepared are organized?

Answer to these types of questions leads to better planning. We will try to answer the questions in a general form, which forms different steps for successful Web site planning. These steps can be applied in same manner to any specific type of the Web site with little care.

Objective and Goal

There is always an objective behind every activity. It is the case for Web site development. Many times activity fails or desired results are not achieved because of unclear objective or purpose. If the objective for developing the Web site is clear, rest of the steps in planning will become easier, otherwise it may create many issues. The Web sites are developed for different reasons. It may be to provide information and better services to customer, to update the members in a community, to make citizens aware of government rules and regulations, to increase the sell or to make the business online. Once the objective is clear, the other activities in planning remain focused around it.

Goal means the achievement from the development of the Web site. The kind of benefit an individual or an organization gets from it is called value addition. It again depends on the reason for the development. If a site is developed to provide the information to the users, the goal should be user satisfaction in terms of getting timely and accurate information. If it is business Web site, goal is to increase in the sell.

CamScanner

For product support Web sites, better services to the customer and thereby retaining customer is the goal. For educational Web sites, it is improved communication and the improved quality of education.

Both objective and goal of the Web site development should be in clear, well-documented form and rest of the steps must match with them. It is also important to define the quantifiable measure to find the success of the Web site. For example, you can count the number of visitors in the predefined period after the launch of the Web site.

Audience/User Profile

Next step in planning is to 'identify the target audience', i.e. the people for whom the Web site is to be developed. It is very important to know about their background, education, gender, age, likings and preferences. This will lead us to choose not only the look and feel of the Web site, but also make it simple to choose the contents and their organization as well as navigation styles. Asking the questions such as – Who are the visitors? Why do they visit the site? How frequently do they visit? What are their expectations, likings and preferences? – give us enough details about the users or visitors. Following are a few examples explaining how user profile affects the design of the Web site.

1. For children Web site, we can use cartoons and graphics with colors. Figure 7 shows home page of www.kidsites.com.
2. For users who are technology savvy, we can use advanced technology and features, as they are likely using latest software.
3. For users interested in music, we can use background music.
4. For the common users, we have to restrict to simple and easy organization of contents so that they can find desired content with little effort.

For continuous improvement, a feedback form is designed to get their requirements and views about the site. This feedback can be used to improve the overall appearance of the site, presentation of the content and navigation styles.

Identifying and Organizing Contents

The basic objective for any Web site is to provide the information in one or other form such as text, image, audio, video, various forms etc. to the target audience. This information forms the contents of the Web site. The content of the Web site is collected from the various sources depending upon the objective of the Web site and user needs. For example, the contents for the Web site of tax department are the rules and regulations and the various forms to download which the citizens can fill and submit. For online tutorial on the given subject area, the content is written by an expert. Regardless of how content is prepared or collected, following characteristics are important:

1. Contents include text, graphics, forms, sound and video.
2. Content should be unambiguous, clear, spell checked, precise and accurate.
3. Content should be relevant, recent and matching to site objective.
4. Content should fulfill the information need of the users.
5. Content should be well organized.

Once the information is identified and collected, the next important activity is to prepare a storyboard that includes the page layouts, their logical grouping and the navigation choices. Either paper-based approach is used or we can use the flowchart software to prepare it. There are various information structures also known as information organization. We can use particular structure depending upon the contents, user needs and navigation choices. A general information structure is shown in Fig. 8.

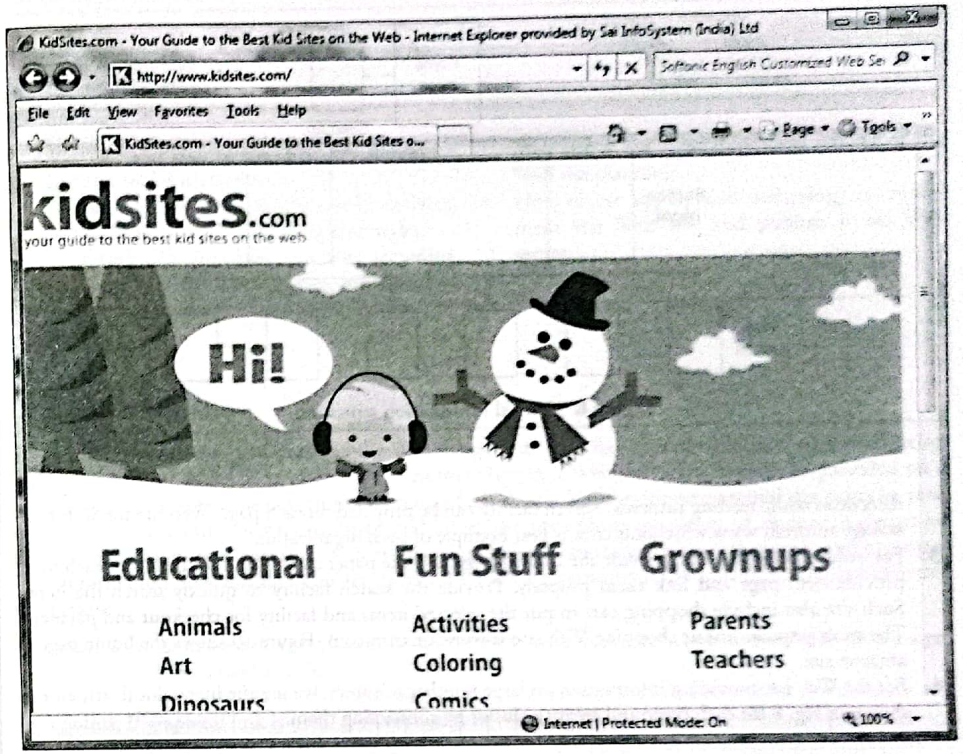


Figure 7 Page from children Web site.

As shown in the figure, the home page is entry to your Web site and it gives links to the various sections. Each section page provides links to the Web pages within the section. For the larger Web site, this hierarchy can be extended to one or more levels. Sitemap provides the link to all the pages directly and it is linked to each page so that user can use Sitemap to visit any part. For the Web site with large information base, search option is provided on the top of each page to search the desired information quickly. Navigation menu on left-hand side or navigation bar on the top helps user move to desired section directly.

The above general structure can be modified as per the type of the Web site and the contents. Following are some examples:

1. For the Web structure, provide free movements within the pages by providing the links between pages. For online free encyclopedia site, wikipedia is best example of such structure.
2. For online tutorial Web site, home page lists all the topics. Once the topic is chosen, a table of contents page or topic list as navigation menu is provided on first page and then sequentially or randomly each page can be visited. Each page of tutorial provides the 'previous' and 'next' link to move in both

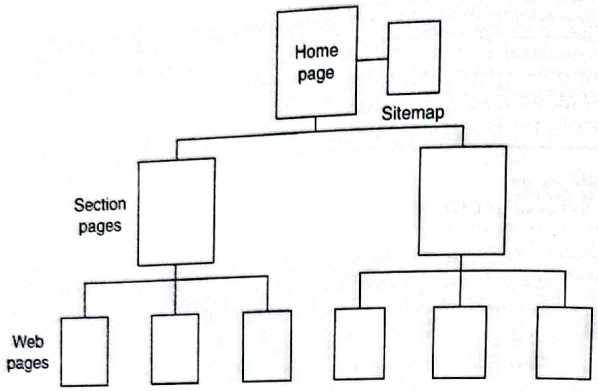


Figure 8 General information structure.

directions while reading tutorials. Search facility can be provided on each page. Web site for Web technology tutorials www.w3schools.com is best example of such organization.

3. For online shopping sites, provide the catalog page just like paper catalog and for details of each item, provide item page and link them properly. Provide the search facility to quickly search the items. Such site also include shopping cart to put the selected items and facility for checkout and payment. The most popular online shopping Web site is www.amazon.com. Figure 10 shows the home page of amazon site.
4. For the Web site providing information on large number of topics, we use the hierarchical structure as shown in Fig. 8 for each topic and create a cluster by connecting them.

Towards the Publishing of Web Site

After the storyboard, following are the steps to be performed before the Web site is publically accessible:

1. Implement each page using various suitable technologies. HTML, XHTML, CSS and JavaScript are client side technologies, whereas Servlet, JSP, ASP, PHP, Perl are the server side technologies.
2. Organize the files containing your Web pages and other contents like images into proper directory structure so that future modification becomes easy.
3. Test your individual pages and their links on local machine so that once launched it will work properly.
4. Register your domain name, e.g. www.myweb.com with appropriate ISP providing hosting Services.
5. Upload the Web site on Web server by copying your folder containing whole Web site into Web server's public area linked to your domain name.
6. Access your Web site from any browser with URL www.myweb.com.

Checkpoint!

We now know that:

- Objective and goal of the Web site should be clear and well-documented.
- User profile plays an important role in deciding look and feel, contents and navigation styles.
- Identifying and organizing contents according to type of Web site, i.e. preparing storyboard of Web site is the most important activity of the Web site planning.
- How to use appropriate technology to implement the Web site and publish it on Web server.

2.4 Navigation

The most important design element in the Web design after page layout is navigation design. Navigation means the ways to move from one page to another page in a Web site using hyperlinks provided on a page. If navigation design is not proper then user feels the problem in moving around the pages in your site in a desired manner or gets confused and leaves the site. There are plenty of choices to design an effective navigation for a Web site. Whatever is your choice, keep information architecture and organization, information flow and user choices in mind. Following are some important tips to design effective navigation:

1. Navigation links are either text based, i.e. a word or a phrase is used as a link, or graphical, i.e. a image, a icon or a logo is used as a link.
2. Navigation links should be clear and meaningful. Use the words related to section or information it points. If graphical link is used, picture should be meaningful.
3. It should be consistent, i.e. use same types and colors on each page. Put the same link at same place on a page. For example, 'Sign out' link can be placed on each page on top right-hand side with same color and type. See the sign out link on Yahoo Web site.
4. Link should be understandable. Link itself guides the user about the contents.
5. Organize the links such that contents are grouped logically.
6. Provide search link, if necessary, usually on top of the page. Use some common links such as 'About us' or 'Contact us'. Such links should appear on each page at same place, normally in navigation menu or horizontal navigation bar.
7. Provide the way to return to first page from anywhere using link 'Home' so that in case of confusion user can start from home page.
8. Provide the user with information regarding location, i.e. where is he/she? Figure 9 shows the page from www.amazon.com with line 'Books -> New & Used Textbooks -> Engineering' indicate where you are.
9. For a Web site having many clear sections, provide navigation menu on left-hand side to directly move to a particular section. Figure 10 shows the home page of www.amazon.com, a popular e-commerce Web site with navigation menu.
10. Horizontal navigation bar can be provided on each page to directly jump to any section from anywhere. Figure 11 shows the navigation bar in the science magazine site for major sections.

CamScanner

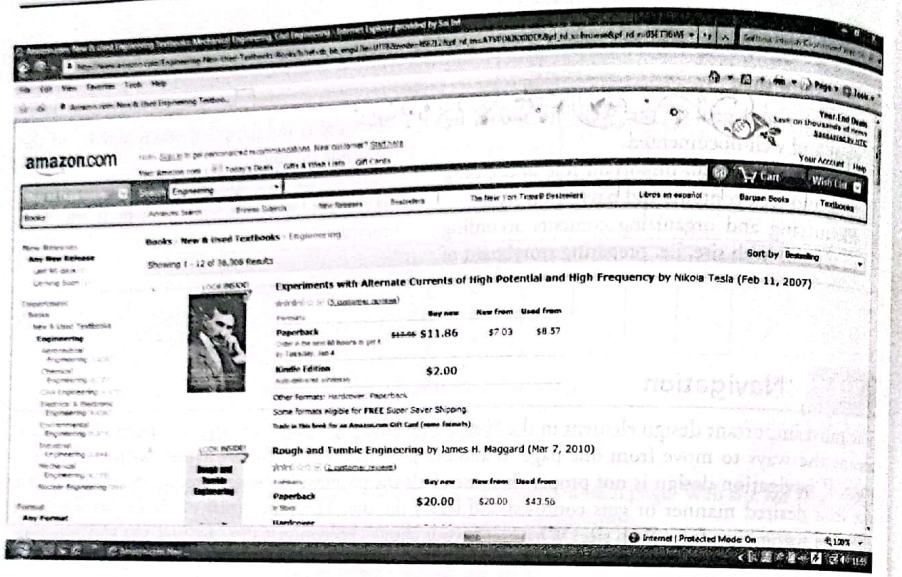


Figure 9 Location information on amazon.

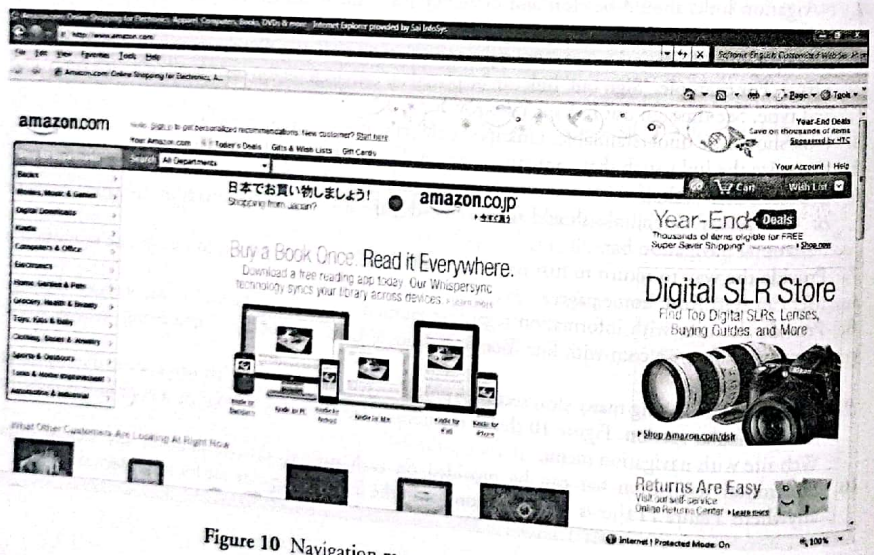


Figure 10 Navigation menu on amazon.

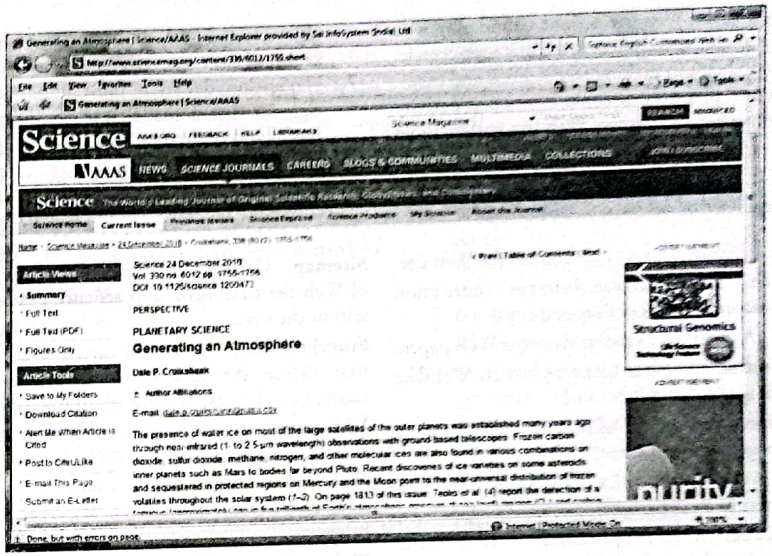


Figure 11 Horizontal navigation bar on Science magazine.

Checkpoint!

We now know that:

- navigation provides the different possible ways to move from page to page in a Web site and it is most important Web site design element.
- keep information organization and flow in mind while designing navigation for the Web site.
- navigation links can be text or graphics based and should be clear, meaningful and consistent.
- provide search option on all the pages, if required and common links such as "About us" and "Contact us".
- provide way to move to home page from anywhere. The location information is very useful.
- use either navigation menu or navigation bar if Web site provides more section or information to group them logically.

Summary

1. Web site is a collection of interlinked Web pages and is a part of large WWW space on Internet.
2. Web site design is a critical issue for success and needs to consider environment issues such as browser support, bandwidth of the connection, cache support and display resolution of the monitor.
3. Look and feel of the Web site forms the overall appearance of the Web site, therefore Web site theme, fonts, graphics and colors need to be carefully chosen. The presentation of information should be clear and easy to access.
4. Page templates are used to design individual pages using grid-structure and distribute the content for quick view and easy access.

CamScanner

5. Design of the Web site is done considering user requirements and choices to make it user-centric. Sitemap helps in making access quick and easy.
6. Web site planning is most important activity in preparing an effective Web site and includes objective and goal, target audience and organization of contents.
7. Navigation design is most important design element that is made clear, consistent, meaningful and understandable.

Key Words and Phrases

Audience People who will visit or use the Web site.

Bandwidth Speed of the Internet connection usually measured in bits per second (bps).

Browser A client software to view the Web pages.

Domain name Unique name by which Web site is accessed.

Information structure Defines the relation among the Web pages within the site.

Link It is a text or graphics pointing to other Web page referred to as hyperlink.

Look and Feel Overall appearance of the Web site is called look and feel of the Web site.

Navigation Defines the various ways to move from page to page within the Web site.

Objective and Goal of Web site Reason or purpose behind developing the site and the value addition to be achieved through the Web site.

Page layout A template of the Web page defining the visual structure of a page.

Resolution Number of pixels on a screen written as no. of columns × no. of rows, i.e. 800 × 600.

Sitemap Graphics representation of the structure of Web site to directly jump to any section or page within the site.

Storyboarding Important activity in planning that define the layouts of the pages and the navigation choices, i.e. flow of information.

Web page A page written in HTML and linked with other pages using hyperlinks.

Web server It is a software that gets the request in form of URL and deliver's the Web page to the Web browser.

Web site It is a collection of Web pages connected together using hyperlinks and accessible using a single domain name.

Web site planning Set of activities to be performed prior to implementation of Web site including objective and goal, audience, contents and its structure, and directory management.

Web site theme It is used to convey the message of the Web site. It may be color scheme or graphics.

Multiple-Choice Questions

1. The best way to make your Web page portable across browsers is to
   (a) Test it on most browsers
   (b) Use features which are common to all the browsers
   (c) Follow the standards
   (d) All of the above
2. If a Web page using CSS is not loaded by the browser properly then it is
   (a) A browser compatibility issue
   (b) A validation problem
   (c) A standard problem
   (d) None of the above
3. Look and feel of the Web site means
   (a) Web site theme
   (b) Web site design
   (c) Overall appearance of the Web site
   (d) All of the above
4. Web typography expresses
   (a) Structure
   (b) Emotion
   (c) Both
   (d) None
5. What makes the presentation of the Web site effective?
   (a) Visual structure of the Web pages
   (b) Navigation links
   (c) White spaces
   (d) All of the above
6. Which of the following technique is used to design page layouts?
   (a) Table tag of the HTML
   (b) CSS
   (c) Any
   (d) None
7. Which of the following criteria is important to make Web design user-centric?
   (a) Whether a page is designed for scanning
   (b) Whether a page is designed for reading
   (c) Organization structure of the information
   (d) All of the above
8. If _____ step is well defined, rest of the steps for planning a Web site are clear and easy to follow.
   (a) Objective and goal
   (b) Audience profile
   (c) Contents of the site
   (d) None of the above
9. What is most important to publish your Web site?
   (a) Web site design
   (b) Use of latest features
   (c) Registering domain name
   (d) Organizing files
10. For easy access and surfing of Web site for average users, which of the following design issue should be strong?
   (a) Page layouts
   (b) Navigation
   (c) Look and feel
   (d) Presentation

Review Questions

1. List the various Web site design issues. Discuss the design issues related to software and hardware environments.
2. What do you mean by browser compatibility issue? What are the possible solutions?
3. Discuss the effect of Web browser, bandwidth of the Internet connection and cache on the design of the Web site.
4. What are the different choices for a Web designer considering the resolution of the display devices?
5. What do you mean by look and feel of the Web site?
6. What is Web site theme? How can you choose a Web site theme? Give some examples.
7. List the important points to be kept in mind while selecting the fonts, graphics and colors for a Web site.
8. What are the important issues to be considered for making presentation of the Web site effective?
9. What do you mean by a page layout? What are the technologies available for preparing page layouts? List the design points to be considered while preparing page layouts.
10. Is location of information on Web page important? Why? Explain with example.
11. What do you mean by user-centric design? What are different techniques to design user-centric Web site?
12. What is site map? What is its purpose?
13. List and explain the different steps for planning a Web site.
14. What do you mean by objectives and goals of a Web site? List the objectives and goals of a Web site for an educational institute.
15. What is included in contents of a Web site? What are the different ways to organize the contents of a Web site?
16. How does knowledge about audience or users of a Web site helps in making Web site successful?

CamScanner

17. What is navigation? What are different navigation styles used by modern successful Web sites?
18. List the important tips to design effective Web site navigation.
19. Give the characteristics of effective navigation.
20. What is navigation bar? How does it allow a user to obtain effective access of the Web site content? Give an example.

Practice

1. Prepare a detailed report for the Web site design issues for each of the following types of Web sites.
   (a) E-commerce
   (b) Education institute
   (c) Community
2. Study the various types of organization of Web site contents from different online resources.
3. Decide at least five important Web design parameters and rate the five different Web sites you are using frequently for these parameters.
4. Survey the Web sites that are very strong in terms of page layout, linking and navigation. List the reasons for your choice.

Chapter Quiz

1. World Wide Web Consortium publishes the various standards for the Web. Visit the Web site of the W3C and make a detail study of the Web Content Accessibility Guidelines (WCAG).

Useful Web Addresses

http://www.w3.org/
http://en.wikipedia.org/

Answers

1. (c)
2. (a)
3. (c)
4. (c)
5. (d)
6. (c)
7. (d)
8. (a)
9. (c)
10. (b)

3 The Client Side: HTML

LEARNING OBJECTIVES

At the end of this chapter, you will be able to:

- List and understand the steps of Web development.
- Understand the structure of the HTML document.
- Display the Web pages written in HTML in your browser.
- Describe and use common HTML tags, hyperlinks, lists, tables and images.
- Write an HTML form to get the input from user.
- List and define various input types for the HTML form.
- Manage the Web site structure in terms of their file organizations.

This chapter starts to look at the client side and static Web page development using HTML. You will learn how to develop simple Web pages and formatting, together with tables, images and frames. Showing the basics of client side, static Web page development using HTML, this chapter describes how to manipulate output with styles, emphasis and fonts. It explores how to embed images into your Web pages along with hyperlinks, which are of prime importance to how the Web works.

This chapter covers the development cycle of producing your Web sites and applications right from the beginning:

1. HTML tags, attributes and document structure.
2. Adding emphasis, styles and fonts.
3. Using color.
4. Commenting and readability of HTML.
5. Pre-formatted text.
6. Hyperlinks.
7. Definition, ordered and unordered lists.
8. Tables.
9. Images.
10. Simple forms.
11. Web site structure.

3.1 Introduction

A Web site is generally composed of individual pages that are linked together, each of these relating to a different aspect of your site, such as news, links and biography. It is possible to try other ways of breaking your site down but at this stage it is best to consider it in terms of pages.

CamScanner

To begin with you don't even need a server to put your pages on; you can just develop on your own computer and view any pages you make with your Web browser.

3.2 The Development Process

When developing there is a cycle that takes place, which generally matches the usual cycle within software development: design, write code, test. For Web development it's probably closer to: study the requirements, design, write the HTML (or script), upload, test and go through it again.

Requirements

It's important to make sure you understand what you are being asked to do before you start! This can be seen from the point of view of both graphic design and functionality. The best idea is to make at least some sketches at the start to play with ideas, then see if you can work out which pages should link to each other.

Design

The design stage tries to provide a solution to match the requirements. It should also take into account what is possible with the various technologies that are available. Sometimes hand coding, although more precise, can simply not be fast enough to meet a deadline! In these cases you may be able to get help from applications such as Dreamweaver. Web sites can require a large amount of work with art and graphics packages as well as code tools.

Write Code

This is the point where you start writing your HTML. Hand coding is slow but precise and usually easier to follow. Generated script and HTML from an application can sometimes also be non-standard!

Test

After spending some time working on your masterpiece you will want to see what it looks like. If you develop using an editor, then the next stage will be to point your browser at the page. If you work with an IDE (Integrated Development Environment), then it's possible that it will show you what a page looks like simply by clicking a button! If your browser is already on the page you can press refresh for it to reload.

Upload

When you are happy with your page you will want to upload for public viewing!

Re-Iterate

The next stage isn't so much a stage, as an instruction to do the above again until you've got it right and it works without bugs.

Checkpoint!

We now know:

- To develop we need an editor, a browser and if we want to upload to a server then we also need an FTP client.
- There is a cycle involved in the process of developing Web pages and applications. This involves designing and thinking about the requirements, writing the HTML, uploading (or simply viewing locally), then viewing with the browser which may simply mean hitting the refresh button if it is already pointing at the site or page.
- The cycle starts again with any changes we want to make after viewing.

3.3 Basic HTML

HTML can be considered as the main language of the Web in some respects. All browsers understand it and because of its simplicity it's generally not taxing for the computers involved. Here we look at initially developing static pages; that is, ones that do not change depending on user input or interaction.

The main aim of HTML is to format a Web page hopefully in the same way for every browser, although it doesn't always work out that way in practice.

Loading Pages with the Browser

A page can be loaded into a browser in a couple of ways: by writing in a URL in the address bar or by going to the menu and clicking on 'file' then 'open' if it's a local file.

If you point your browser at a directory on a Web site it will try and find a series of files beginning with `index.html`, then usually `index.htm`, `index.shtml`, `index.php` and so on. Generally when you are browsing it's best to write in an address ending in a directory with a trailing forward slash otherwise the browser thinks it is a file and tries to load it, before correcting itself. Don't add the forward slash to URLs ending in a real file though!

A Page on the Web

How do we start? Generally if you point a Web browser at a document it will do its best to read it. For example, if you use 'open' on the file menu of your browser and enter the name of a local file called `hello.txt` containing some simple text:

```
Hello there!
```

It will actually read and display it.

HTML Document Structure

To make a 'real' Web page we need to add some elements and rename the file to either htm or html. Open a new blank file in your editor and write some code in:

```
<html>
     A little bit of hypertext
</html>
```

Anything written in triangular brackets is known as a tag and is part of the markup language. So, in this small page we have a number of tags but what do they mean? The first thing to look at is the `<html>` tag, which simply identifies this as HTML. Notice here that the first line and the last line are both `<html>` tags except the last one contains a slash, meaning that this one is a closing tag and matches the tag at the top. Most HTML follows this convention, although there are several tags that don't and you can get away with just the one tag, as an instruction. Also notice that the HTML tags are in lowercase. Although HTML tags are not case sensitive, the World Wide Web Consortium (W3C) suggests the use of lowercase tags in their HTML 4 recommendation.

The `<html>` tag identifies a section of HTML code, opening with the `<html>` and closing with `</html>`. Everything between these is the code itself. Here we find a simple text message, which is output straight to the browser screen:

```
<html>
     <head>
          <title>
               The amazing art of Web programming
          </title>
```

CamScanner

```
    </head>
    <body>
        a little bit of hypertext
    </body>
</html>
```

Our next, more properly formed HTML Web page extends the first, adding a title to the page and dividing the document up into head and body sections. The body section contains the main document while the head contains details such as the title as shown here. All full HTML documents should follow this structure and include a body and head.

An HTML element then begins with a start tag and ends with a closing tag:

```
<title> The amazing art of Web programming </title>
```

In our simple Web page, there is also the tag, <body>. This identifies the next part between <body> and </body> as the main section. This contains the code for the Web page document.

3.4 Formatting and Fonts

If we want to add more lines into the body section as text to output we can add them with our editor:

```
<body>
    a little bit of hypertext<br>
    makes the world go round<br>
    and <i>around</i><br>
</body>
```

This body section contains several new tags. The <br> or break tag makes output start on the next line down at the beginning of a new line. Notice how this tag is self-contained in that it does not need a closing tag. If the lines were formatted in the HTML document in some other way, would it matter? Like:

```
<body>
    a little bit of hypertext<br>makes the world go round<br>
    and <i>around</i><br>
</body>
```

The answer to this is no, because formatting is only taken from the tags themselves not the layout of your document. It is still a good idea though to lay out your HTML in a readable manner so you can understand it easily should there be a problem that you need to find. It's quite a good idea to lay it out in a structured way, following indentations that relate to the structure, as is done in the programs here.

Some programs will generate code, allowing you to design your Web page at a 'top level' more like a graphic designer, or simply output your text document, capturing its styles and formatting as well as any images that are embedded. These usually produce almost unreadable and copious quantities of code that are hard to follow and nearly impossible to correct, change or debug by hand.

Another way of adding breaks is to define paragraphs with the tag. This can be used on its own, like the <br> tag or with a closing at the end of the paragraph. The tag places a blank line before the line it is on.

To break sections of a page the tag <hr> can be used; this creates a line or horizontal rule. Like <br> it does not require an ending tag.

Concept Check

- Try writing a Web page in an editor.
- Add line breaks and suitable formatting.
- Save it with an HTML extension.
- Start up a browser and open it with the 'file' command on the menu.
- Is it what you expected? If not go back to the editor, alter it and try again!

Using Types of Emphasis

The previous HTML code used the tag; any words enclosed by the and will be made italic. There are many ways of emphasizing or altering the text you output, as is shown in Table 1.

Pre-Formatted Text

A useful formatting element is <pre>...</pre>, which enables you to embed text that is already formatted so you don't have to put break tags in, for example:

```
<pre>
 This is already set out in the way I want it
 It has some advantages and is quick
 You don't have to add line breaks but
 It hasn't proportional spacing and is in courier font...!
</pre>
```

The text is output in courier font which is not proportional spacing, i.e. all characters take up the same space. This allows you to position text with white space (the space character) if you like, which makes it a little easier to format. Other tags can still be embedded inside to help with formatting headings, font sizes and aligning.

Table 1 Adding emphasis

Tag	*Format*
...	Italic
...	Bold
<tt>...</tt>	Typewriter effect
...	Emphasis
<blink>...</blink>	Blinking
^{...}	Superscript
_{...}	Subscript

CamScanner

Table 2 Font comparison

Font Size	*Heading*	*Point Size*
7	–	36 pt
6	<h1>	24 pt
5	<h2>	18 pt
	<h3>	12 pt
4	<h4>	12 pt bold
3	body text	12 pt plain
	<h5>	10 pt
	<h6>	7 pt
2	–	9 pt

Font Sizes

You may also want to change the size of your text. There are a few ways of doing this too. You can use the `<font>` tag or the `<h...>` tags. As you can see from Table 2 the `<h...>` tags don't just control the size but some degree of emphasis for headings.

The size of your text can also be changed with the `<font size=...>` tag and associated end tag `</font>` with the font size running from 2 to 7. The ending font tag will make the font revert to the previous font used.

You can align your text to `left`, `right` or `center` by combining one of the above tags with the align markup:

```
<html>
    <head>
        <title>
            the amazing art of Web programming
        </title>
    </head>
    <body>
        <p align="center">a little bit of hypertext<br>
        makes the world go round<br>
        and <i>around</i><br></p>
    </body>
</html>
```

This will make all the text be centered in the current document until the ending paragraph tag, `</p>` is met. The align tag is added as an option to the main function. You can also put these optional align statements into heading tags `<h...>`.

```
<html>
    <head>
        <title>
            the amazing art of Web programming
        </title>
    </head>
```



```
    <body>
        <h1 align="center">hypertext alignment</h1>
        <p align="center">a little bit of hypertext<br>
        makes the world go round<br>
        and <i>around</i><br></p>
        <p align="left">to the left...</p>
        <p align="right">...and to the right</p>
    </body>
</html>
```

Concept Check

- Write another Web page, utilizing font changes, page breaks, horizontal rules and alignment of text.
- Write a Web page using the `<pre>` element. Experiment with layout, compare with using HTML formatting tags to do the same thing.

3.5 Commenting Code

When the HTML code starts building up and looking complicated it's probably time to add commenting. You can do this using `<!--...-->`:

```
<!-- This is a complex part of my Web page, it adds a border and
        several types of font -->
```

Later on, when you have forgotten what you have done in a Web document this can help you understand quickly the various sections of your nicely structured HTML. It's also good from the point of view of other people understanding it who may take over the responsibility of maintaining a Web site, after you have gone.

3.6 Color

Adding color (the right color, that is!) to a Web site is an important part of both the design and coding. The two main basic parts to add color through HTML are the background of the browser display and the text. You can use:

```
<body bgcolor = value text = value>
```

which will set both within the body section of the document. If you want to just change the color of the text it's possible to use:

```
<font color = value>
```

which is similar to the optional `size` attribute that can be added. The actual values involved are based on a red, green, blue (RGB) scheme, usually entered as hexadecimal numbers. Hexadecimal is the number system that runs from 0 to 9, then through the letters A–F, as shown in Table 3.

CamScanner

Table 3 Number base conversion

Base 10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13

If you want to convert from hex to decimal, multiply the first digit (in the case of a two digit number) by 16 and add the last number to it. For example, if we have the hexadecimal number A2 then we multiply 'A' (which is 10) by 16 and add in 2. So in this case A2 hex = 162 in base 10.

With two digits the maximum number that can be represented in hexadecimal is FF, or 255.

To set a color value for HTML you supply six digits, representing the RGB color proportions you want; a guideline is shown in Table 4. The value then is actually composed of three numbers, each of two digits. So, the following would set the color to red for the background:

```
<body bgcolor="#800000">
```

whereas

```
<body text="#ffff00">
```

will set the main document's text to yellow.

In theory, any color you want can be made from the red, green, blue palette by combining different amounts of the colors in the right proportions.

Table 4 Some example colors with codes

R	*G*	*B*	*Color Produced*
00	00	00	Black
FF	00	00	Bright red
00	FF	00	Bright green
00	00	FF	Bright blue
80	00	00	Dark red
00	80	00	Dark green
00	00	80	Dark blue
FF	FF	00	Bright yellow
80	80	00	Brown
FF	00	FF	Magenta
80	00	80	Indigo
00	80	80	Turquoise
80	80	80	Grey
FF	FF	FF	White

Concept Check

- Try out using colors; can you make each letter of a word a new color, giving a rainbow effect?

3.7 Hyperlinks

Hyperlinks are one of the strongest aspects of why the Web came about in the first place – creating the ability to associate information between resources. The way to form a link is fairly simple:

```
<a href = "http://www.someWebpage.com">the text link</a>
```

The actual hyperlink tag is `<a href ...>`, the Web address in quotes is where it goes if clicked on and the text afterwards is what is printed on the screen to select. Finally, the link ends with a closing tag. The `href` stands for Hypertext REFerence, the a stands for anchor because the link can't be used alone; it must have either an image or text to anchor to. The link formed is a URL, which can be any of the usual kinds, including FTP.

Sometimes it's useful for a selected link to open in a new browser window rather than using the current one. To do this the `target` attribute is used:

```
<a href="http://www.someWebpage.com" target="_blank">the text
        link</a>
```

Links can also be made to places within an already loaded page. This is done by creating an anchor point to the place you may want to jump to and a corresponding link:

```
<html>
    <a name="thetop">introduction</a><br>
    this could be a long document!<br><br><br><br>
    <hr>
    this may be some way down the page.<br><br>
    complete with a link to
    <a href="#thetop">back to the top</a>
</html>
```

The `<a name="...">` creates the anchor point for the link to jump to, while the `<a href="#...">` is the link. It's also possible to form a link from another page to an anchored point. For example, if the page above was called `leaparound.html` then the link

```
<a href="leaparound.html#thetop">
```

would point the browser at the anchored point on that page.

It's possible to reference sub-directories under the current working directory by using links such as

```
<a href="images/seaside.jpg">my day out</a>
```

If you are in a sub-directory and want to link into a file above that, then use the two dots:

```
<a href="../mypage.html">up to that other page</a>
```

CamScanner

Checkpoint!

We now know:

- The basics of making a Web page and displaying it with the browser.
- The structure of a Web document.
- How to output text and add emphasis.
- How to control text font size.
- The basis behind using color.
- How to add commenting to make HTML easier to understand.
- How to create hyperlinks to other documents and places within the same document.

3.8 Lists

A useful ability in HTML is to form quick, concise lists, which can be numbered or just bulleted.

Unordered Lists

A simple bulleted list can be made with the . . . tags. For example:

```
<html>
    <head>
        <title>
            the amazing art of Web programming
        </title>
    </head>
        <body>
            <ul>
                <li>Life of Pi
                <li>Sophie's World
                <li>Jonathan Strange and Mr Norrell
                <li>The Sandman Endless Nights
            </ul>
        </body>
</html>
```

will produce a list of book titles that is bulleted with round points. The actual character used can be changed and set to `square`, `disc` (default) or `circle`. To do this, simply give the option to the `<ul>` tag:

```
<ul type="square">
```

Ordered Lists

It's also possible to have ordered lists using `<ol>...</ol>`:

```
<ol>
    <li>Life of Pi
    <li>Sophie's World
```



```
    <li>Jonathan Strange and Mr Norrell
    <li>The Sandman Endless Nights
<ol>
```

Again, there are options available. In this case it is possible to have different numbering rather than the default 1, 2, 3, 4, . . . :

```
<ol type= i>
```

where the various types are:

1. i Roman numerals i, ii, iii, iv
2. I Roman capitals I, II, III, IV
3. a Lowercase letters a, b, c
4. A Capital letters A, B, C.

Nested Lists

You can also nest one list within another, so you could make an unordered list inside a numbered one, or vice versa:

```
<html>
<h2>To Do List</h2>
<ol>
    <li>clear out garage
    <li>prepare report
    <li>write email to Ray
    <li>go shopping
        <ul>
            <li>tomatoes
            <li>pasta sauce
            <li>vegetarian cheese
            <li>long spaghetti
        </ul>
    <li>repair fence
</ol>
</html>
```

In this example, we have first an ordered list, then switch to yet another list under `go shopping` in which there is a list of items to buy. Once that ends the old list is resumed. You can also change the font of one list, or change the emphasis to make it stand out.

Definition Lists

There is also another kind of list that is used for lists of terms and definitions but could be used anywhere you want sets of sub-headings followed by text:

```
<html>
    <dl>
        <dt>Hawthorn Shield bug
        <dd>Beetle which mainly lives on hawthorn bushes
```

CamScanner

```
            <dt>Minstrel bug
            <dd>A black and red striped beetle
            <dt>Green Shield bug
            <dd>Large beetle common in UK gardens
        </dl>
    </html>
```

3.9 Tables

Often we need to show data in tabular form that is, an area that is split into columns and rows to show associations, for example. In HTML this is fairly easy to achieve and requires only a few tags.

Table Structure

A table starts with a `<table>` tag and ends with `</table>`. Each row begins with `<tr>` and ends with `</tr>`; between lie the columns defined with table definition tags `<td> . . . </td>`:

```
<html>
    <table>
        <tr>
            <td>column item 1</td>
            <td>column item 2</td>
        </tr>
        <tr>
            <td>another item</td>
            <td>and another</td>
        </tr>
    </table>
</html>
```

You may want to group your data with associated headings, for example:

```
<html>
    <table border>
        <tr>
            <td><b>Egyptian</b></td>
            <td>Osiris, Isis, Horus, Thoth, Ptah, Hathor </td>
        </tr>
        <tr>
            <td><b>Norse</b></td>
            <td>Odin, Frigg, Thor, Loki, Freya</td>
        </tr>
        <tr>
            <td><b>Greek</b></td>
            <td>Zeus, Apollo, Persephone, Hermes, Hera</td>
        </tr>
        <tr>
            <td><b>Roman</b></td>
```



```
            <td>Jupiter, Luna, Mars, Mercury, Diana</td>
        </tr>
    </table>
</html>
```

As you can see from this example the table now has a drawn border on all sides and the columns and rows. This is defined by using the border option inside the table tag to increase the width of the border shown; for example, `border=n` where n is the thickness in pixels.

Table Headers

It's also possible to create a table with properly defined headers:

```
<html>
    <table border>
        <tr>
            <th></th>
            <th>Green Park</th>
            <th>Oxford Circus</th>
            <th>Euston</th>
            <th>Finsbury Park</th>
        </tr>
        <tr>
            <th>Tube</th>
            <td>5mins</td>
            <td>8mins</td>
            <td>10mins</td>
            <td>13mins</td>
        </tr>
        <tr>
            <th>Bus</th>
            <td>12mins</td>
            <td>14mins</td>
            <td>18mins</td>
            <td>20mins</td>
        </tr>
    </table>
</html>
```

This table is set out with a necessary blank cell as a spacer for the titles in the columns and rows; the HTML is simply a header with no entry. Figure 1 shows this table.

Irregular Tables

The kinds of table mentioned so far are relatively simple and symmetrical but sometimes it's necessary to make irregularly shaped tables:

```
<html>
    <table border>
```

CamScanner

```
<tr>
    <th rowspan=2></th>
    <th colspan=4 rowspan=1>TUBE STATION</th>
</tr>
<tr>
    <th>Green Park</th>
    <th>Oxford Circus</th>
    <th>Euston</th>
    <th>Finsbury Park</th>
</tr>
<tr>
    <th>Tube</th>
    <td>5mins</td>
    <td>8mins</td>
    <td>10mins</td>
    <td>13mins</td>
</tr>
<tr>
    <th>Bus</th>
    <td>12mins</td>
    <td>14mins</td>
    <td>18mins</td>
    <td>20mins</td>
</tr>
</table>
</html>
```

Figure 2 shows the table produced.

This table is irregular because of the added heading within the table. To include this, four columns had to be used and also space added at the start for a block of two rows by two columns. The way this is done is by the optional attributes `colspan` and `rowspan` within `<th>` tags, describing how many columns and rows it should take up. These can be added to `<td>` tags as well.

	Green Park	Oxford Circus	Euston	Finsbury Park
Tube	5mins	8mins	10mins	13mins
Bus	12mins	14mins	18mins	20mins

Figure 1 A simple HTML table.

	TUBE STATION			
	Green Park	Oxford Circus	Euston	Finsbury Park
Tube	5mins	8mins	10mins	13mins
Bus	12mins	14mins	18mins	20mins

Figure 2 An irregularly shaped HTML table.

Another formatting option within tables is alignment; this is added using `align` for horizontal and `valign` for vertical alignment. For example, it's possible to center code within a cell using this code:

```
<tr>
    <th>Tube</th>
    <td align=center>5mins</td>
    <td align=center>8mins</td>
    <td align=center>10mins</td>
    <td align=center>13mins</td>
</tr>
```

It's important to remember that tables will adjust to fit any window that's open so such alignment is important and you may want to consider using vertical as well as horizontal alignment.

Tables and Page Layout

Tables can also be used to format the layout of an HTML page. You can easily divide the page into sections and fill the different areas with text, which will be held in place by the table. The table can be invisible, without borders so it does not interfere with the design:

```
<html>
<head><title>table style</title>/head>
<body>
<table border="0" width="100%" cellpadding="10">
<tr>
<td width="50%" valign="top">
This is the left column containing some interesting text.
Dividing the text like this is similar to newspaper columns.
</td>
<td width="50%" valign="top">
This is the right column which contains yet more text.
This simply fills the right side of the page.
</td>
</tr>
</table>
</body>
</html>
```

The attributes used in the table in this case are as follows:

1. `Border`, which sets the border width in pixels around the table
2. `Width`, which in this case is as a percentage of the screen
3. `Cellpadding`, to give the distance in pixels between the inner border and the text.

Another optional attribute is `cellspacing`, which sets the spacing in pixels between the inner and outer borders. The data cells within the table are defined by the `<td>` tag, which also in this case uses `width` to set the width of the cell as a percentage of the browser window. The `valign` puts any data at the top, middle or bottom of the cell.

Some complicated layouts can be made using tables to control the placement of items.

CamScanner

Concept Check

- Experiment with building tables in HTML by drawing up your weekly schedule or timetable.

3.10 Images

As well as text, HTML can be used to place images within pages. This is done using the `<img src="filename">` tag, which, if used on its own, would place the image aligned to the left.

Positioning and Placing Images

```
<html>
<h2>show an image</h2>
<img src="photosnap.gif">
my holidays!
<p>the snow slopes where I went skiing...</p>
</html>
```

This would place the image `photosnap.gif` on a page with the writing straight after it. The graphic should be in the same directory as the HTML file, or directed to the file using the path.

You can use the option `align` within the image tag to make any accompanying text wrap around the image at various positions. Usually the text will be placed in line with the bottom of the image. Any long text following an image, if there is enough space, will fit in the remaining space, wrapping on to the lines below. The `align` options are `bottom` (default), `middle` or `top`.

The image can be set to the middle of the page by using the `<img src>` tag, wrapped by a `<center>` tag.

It's often useful to place a border around an image to space out surrounding text. This is done by using the options `hspace` (setting space left and right) and `vspace` (setting space above and below the image). The amount you set is in pixels. The border is not marked with lines unless you want it to be, using `border=n`.

Resizing an Image

Images can be manipulated to set them to the size you want them to be for the layout you are using, so it's possible to increase or decrease their size in the X and Y directions. The size can be set in pixels or as a percentage of the browser display. The fixed method relies on the options `height` and `width` within the `<img src>` tag; for example, if you want to set a size of 100 by 100:

```
<img src="snapone.gif" height=100 width=100>
```

On the other hand a relative variable setting will sometimes be more useful:

```
<img src="snapone.gif" height=50% width=25%>
```

The height in this case will be 50% of the browser height by 25% of the current browser width. Just setting one of these values will make it be applied to both.

Sometimes an image may fail to load for some reason, or image loading may be switched off or not available. You may want your visitor to still know what was on the page and missing; the way to do this is by using the `alt` option attached to the `<img src>` tag:

```
<img src "me.jpg" alt="my picture!">
```

Background Images

Sometimes you may want the background filling with an image, and this is done using different markup. Instead an option is added into the `<body>` tag as

```
<body background="mypattern.gif">
```

This might be one huge image or it could be a simple picture that can be repeated over the area of the screen, possibly a simple pattern or tile. The rule here is that an image smaller than the window will be tiled (repeated) over the area of the browser screen. Remember, if you want to read text or see other images, then any background image should be fairly simple.

An image may also be used as a link rather than text:

```
<a href="biog.html"><img src="mypic.gif"></a>
```

Checkpoint!

We now know:

- Implement different types of list.
- Utilize tables to give structure to data and layout.
- Embed images and use them as background pictures.

3.11 Simple HTML Forms

Up until now all the HTML that has been dealt with is one way – as output to the browser – but often there is the need to collect information from the user, such as name, address and other contact details. It may be that if your site is an e-commerce site you want to also collect customers' card or payment details along with their choice of product.

Once the user has filled out a form the information needs processing so, on their own, forms don't do a lot! The processing can be done on the client side, using a language like JavaScript, or on the server side with a language such as PHP.

Making a Form

The main tag to build a form is `<form>`. It has a few optional attributes too. Here is an example of the form element:

```
<form action="processform.php" method="post">
</form>
```

The action attribute tells the HTML where to send the collected information, while the method attribute describes the way to send it.

CamScanner

Types of Input

The main tag for collecting information from the user is `<input>`. A simple example here could be:

```
Enter your email address<input name="email" size=35>
```

The text is to give a prompt as to what is expected from the user. The tag itself contains a `name` attribute, so that we can refer to the input by a name, and the size of the entry box in characters:

```
<html>
    <form>
        please enter your user name
        <input name="user" type="text" size=30><br>
        favourite color
        <input name="color" type="text" size=14><br>
    </form>
</html>
```

There are quite a few different types of input to choose from:

1. `<input type="text"/>` This is the default input type and accepts characters and numbers into a text box. It can also have a value attribute attached to it, which will give it an initial value.
2. `<input type="password"/>` This is similar to the above text box but anything that is typed cannot be seen; instead an asterisk is printed to cover up the entry. As the name suggests, this is used for password entry.
3. `<input type="checkbox"/>` This gives a box that can be toggled between checked and unchecked. It can initially be set to one or the other with `checked="checked"`.
4. `<input type="radio"/>` This is similar to the checkbox but in a group of radio buttons only one can be selected at a time. This can also have an initial checked state on one of the radio buttons.
5. `<input type="file"/>` This will give a box to allow you to choose a file similar to when you open or save files usually on your machine. It can be used to select a file on the local machine for upload to a server, for example.
6. `<input type="submit"/>` This allows a form to be submitted. When pressed, the information will be passed on for processing, usually to a script mentioned in the action attribute option of the form.
7. `<input type="image"/>` This will also submit the form when selected and, like the img tag, requires the src attribute to specify an associated image.
8. `<input type="button"/>` This makes a button available.
9. `<input type="reset"/>` This will reset the form to its initial state when selected.
10. `<input type="hidden"/>` This allows hidden data (not seen by the user) to be passed along with the form.

Text Areas

Sometimes you may want to allow the user space to write comments or a message. For this you probably want a larger space for them to write and the best way to do this is to use `<textarea>`:

```
<hr>you can leave a message for me here:
<textarea name="comments" rows="10" cols="70"
wrap="wrap">message:</textarea>
```


Again, the text area is given a name so the information typed in can be referred to when it is being processed. The size of the area is defined by two optional attributes `rows` and `cols` and the `wrap` option provides several ways of letting the wrap within the box: `off`, `virtual` and `physical`. The `off` option stops all wrapping on lines and requires the user to type 'enter' to finish a line. `virtual` wraps the sentence in the box but sends it as one long string and physical provides full text wrapping, with information being kept as it is seen on the display.

Drop Down Menus

Drop down select boxes are another input method that can be used. This is a simple example:

```
<select>
<option value="selected1">Option 1</option>
<option value="selected2">Option 2</option>
<option value="selected3">Option 3</option>
</select>
```

When an option is selected the value returned is set appropriately to `selected1`, `selected2` or `selected3`. The text displayed is the `Option n string`. In a similar way to the `check-boxes` with the `checked` option, the `select` tag allows `selected="selected"` for a default, initial value.

All form values must have a `name` attribute otherwise they cannot be processed properly! Elements can also have a `readonly` attribute, which stops an element from being changed. This may be to confirm past information with new data added, for example.

Listing 1 incorporates the form features that have been explored.

```
<html>
  <head><title>My Form</title></head>
  <form action="processor.php" method="post">
    <fieldset><legend>log in now</legend>
    <b>Please enter your user name</b>
    <input name="user" type="text" size=30><br>
    <b>Your password is </b>
    <input name="pass" type="password" size=10><br><br>
    </fieldset>
    <fieldset><legend>Choices!</legend>
    <b>Favourite color</b>
    <input name="color" type="text" value="green"
                size=14><br><br>
     My favourite fruit is:
    <select>
        <option value="selected1">Apple</option>
        <option value="selected2">Banana</option>
        <option value="selected3">Orange</option>
    </select>
    <br><br>
    I live in a:
    <input type="radio" name="accom" value="house"
                checked="checked"/>House
    <input type="radio" name="accom" value="flat"/>Flat
```

CamScanner

```
<input type="radio" name="accom" value="bedsit"/>Bedsit
<input type="radio" name="accom"
          value="caravan"/>Caravan

<br><br>
My upload file is:
<input type="file" name="upload" size=40><br><br>
</fieldset>
<fieldset><legend>Leaving?</legend>
<b>You can leave a message for me here:</b>
<textarea name="comments" rows=10 cols=70></textarea>
<br><br>
<input type="submit" value="Go!">
<input type="reset" value="Start again"></fieldset>
</form>
</html>
```

Listing 1 Summarizing simple HTML forms.

The form begins at the `<form>` tag with the `action` attribute being pointed at the script, which would process the data supplied by the user who fills the form out. The processing, including the method `post`, will be covered in some detail in later sections on JavaScript and PHP. In this page the form is broken into several sections using the `fieldset` element, which basically draws rectangles around the areas (see Fig. 3). `legend` gives a heading to each input section.

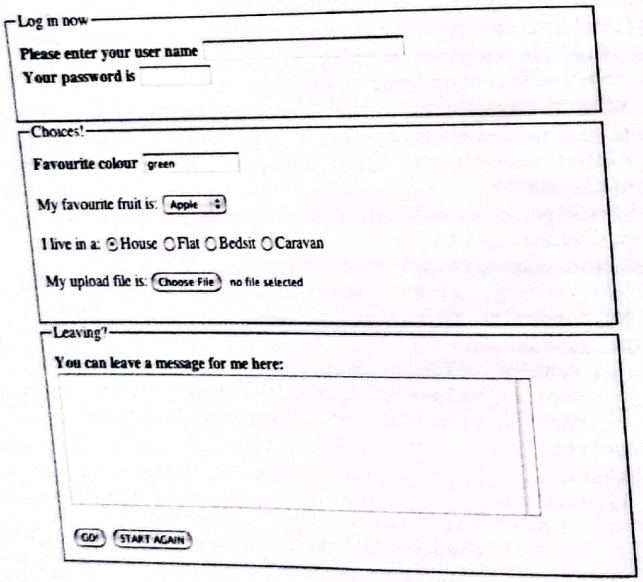


Figure 3 A simple HTML form.

Concept Check

- Create a form to collect details of a user such as name, login, address, telephone number. Also, collect details of items they wish to purchase, such as in a book store, along with a final comments box.
- Notice how there are different ways of inputting the same data for choices: you can use selection boxes or radio buttons, whereas some data is better collected through a text box.

3.12 Web Site Structure

A Web site is a collection of pages associated by hyperlinks, but it should also be broken up into areas for structure, which aids both memory and ordering resources. So, your main site can be broken into several sub-sites. Not only this, but you can separate specific files on their types; for example, graphic files can be kept in their own folder as long as they are referenced properly.

A Web site generally has a root directory, the one which is entered first, then several sub-directories that serve as the sub-sites.

In Fig. 4 the main site has a default index.html page, which will be picked up by the browser simply by going to `http://www.mywebsite.co.uk`. The `index.html` page uses several images that are collected under an images sub-directory. There are two further sub-sites under the main root directory, namely cats and holidays. What would happen if the browser was pointed to the following?

```
http://www.mywebsite.co.uk/cats/
```

An error message would come up, as there is no default page such as `index.html`! So, hopefully there would be a link to the two cat pages from the main `index.html` page. This also applies to holidays, which has a single HTML page and a directory for any images.

By breaking the site into directories it makes it a lot easier to find files and also to navigate the actual Web site, which uses the sub-directories as extensions of the main site.

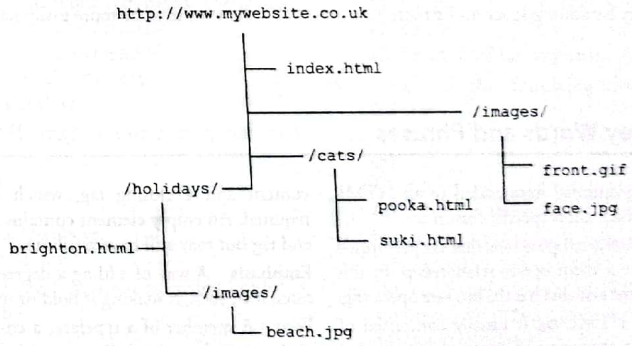


Figure 4 Mapping a Web site.

Checkpoint!

We now know:

- The basics of form development using HTML with various types of input.
- The importance of structuring your Web site properly and breaking material up into sub-directories on the basis of file type and subject matter.

Summary

1. The Internet and the Web in particular can be seen as having applications and events that occur on two sides: the client side and the server side. The client side is the computer with the browser running on it.
2. The development of Web material follows a cycle: requirements capture, design, write code, test, upload, re-iterate.
3. A browser will do its best to load a resource it is pointed at, whether it is a text file, image or HTML page.
4. Properly developed HTML pages have a defined structure.
5. Using HTML a wide range of text and formatting controls exist, including emphasis, fonts, styles, color and layout formatting.
6. You should add comments, particularly in complex documents and endeavour to increase readability by adding space and structure.
7. Hyperlinks can be added to connect documents and resources. Web links can be made to open in a new browser window, if required.
8. Various kinds of list can be created easily and quickly.
9. Tables can be used to format data and control the layout of pages.
10. Images and graphics can be embedded into pages or used as background pictures to make sites more appealing.
11. Forms can be used to gain user interactivity and input of data, which can be processed by scripts. Many types of user input exist as choices for the designer.
12. Care should be taken over the structure of a developed Web site. Sub-directories can be used to sort different kinds of file and also to allow a site to be more easily navigated.

Key Words and Phrases

Attribute An optional item added to an HTML element tag, which alters specific features.

Client side Refers to operations that are performed by the client in a client–server relationship; in this context, this is the side that has the browser operating.

Element An HTML tag is usually comprised of an opening tag consisting of optional parameters/content and a closing tag, which is not always required. An empty element contains no content or end tag but may still have attributes.

Emphasis A way of adding a degree of impact to some text, such as making it bold or italic.

Font A member of a typeface, a co-ordinated set of character designs usually comprising an alphabet, a set of numerals and punctuation, although it may also contain symbols too.

Form A data collection mechanism within HTML that allows the design of various styles of input to suit most types of information.

Heading A set of tags that specify font size and emphasis for different levels within a document.

Horizontal rule A horizontal line that runs across the page; different types can be selected.

Hyperlink A reference to another document or resource.

Line break A formatting term that relates to where a sentence or paragraph is dropped to the next line down. In HTML this can occur when a `<br>` element is used.

List An HTML element that allows the simple presentation of lists of information in unordered or ordered (enumerated) forms.

Point size A relative measure of the size of a font, which used to have a more concrete meaning.

Radio button A type of input mechanism on an HTML form, which usually selects one item from many others.

Server side Refers to operations that are performed by the server in the client–server relationship. Typically it is the machine that runs the Web server or other server software.

Table An HTML element that allows layout and data to be structured.

Tag Part of element in HTML, comprising `<tagname attributes>`; that is, a triangular less than, the tag name and a list of optional attributes. Usually, there is a closing tag such as `</tagname>` but it is not always required.

Text area An input area on an HTML form, which can be sized as required.

Upload To copy or move files to the server from the client. This is usually done as part of the development cycle of producing a Web site or application.

Multiple-Choice Questions

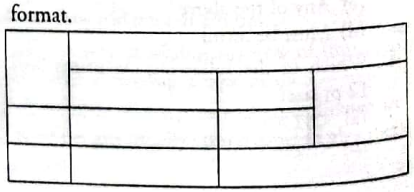
1. Which of the following is true for HTML?
   (a) Used to write the Web pages
   (b) Displayed by Web browser
   (c) Markup language
   (d) All of the above
2. The visible content of a Web page is given inside
   (a) `<html> .. </html>`
   (b) `<head> .. <head>`
   (c) `<body> .. </body>`
   (d) All of the above
3. The HTML page is stored in a file with extension
   (a) .html
   (b) .htm
   (c) Any of the above
   (d) Must be .html
4. Which of the following tag prints header in 12 pt size?
   (a) `<h1>`
   (b) `<h2>`
   (c) `<h3>`
   (d) `<h4>`
5. Which of the following is not used for emphasis?
   (a) `<em> .. </em>`
   (b) `<sup> .. </sup>`
   (c) `<tt> .. </tt>`
   (d) All are used for emphasis
6. Which of the following does not have end tag?
   (a) `<br>`
   (b) `<b>`
   (c) `<p>`
   (d) All of the above
7. Which of the following is correct?
   (a) `<a>Test.html</a>`
   (b) `<a>href="Test.html">Test</a>`
   (c) `<a>url="Test.htm">Test</a>`
   (d) All of the above

CamScanner

8. To open a link in new browser window, which of the following is used?
   (a) target = "_blank"
   (b) target = "_new"
   (c) target = "_window"
   (d) window "_new"
9. Which of the following is not related to <table> tag?
   (a) <table>
   (b) <th>
   (c) <tt>
   (d) <tr>
10. Which of the following tags produces the list with numbers?
   (a) <ul>
   (b) <ol>
   (c) <dl>
   (d) None of the above
11. Which of the following is correct for radio button?
   (a) <radio>
   (b) <input type="radio" />
   (c) <input type="radio-button" />
   (d) None of the above
12. Which of the following is correct for creating drop down list?
   (a) <select> .. <select>
   (b) <input type="dropdown" />
   (c) <input type="list" />
   (d) None of the above
13. The attribute "alt" is used with the _____ tag.
   (a) <body>
   (b) <img>
   (c) <table>
   (d) All of the above
14. Which of the following tag is used to divide the form into sections?
   (a) <break>
   (b) <p>
   (c) <fieldset>
   (d) None of the above

Review Questions

1. Define the terms: Web page, Web site.
2. What is a Web site? Discuss the development life cycle of Web site.
3. What is HTML? Discuss the structure of an HTML document with major sections.
4. What is an HTML tag? Discuss the format of an HTML tag with example.
5. What are the different emphases in HTML? Show their use with example.
6. What is header <h..> tag? Show its use in Web page.
7. What is pre-formatted text in HTML? Give an example.
8. How can you use colors in Web pages? Show with examples.
9. Define the tag font along with important attributes and demonstrate its use.
10. What is hyperlink? How do you define it? Show the various use of hyperlinks.
11. How can you jump to particular point in same or different page using hyperlink? Give examples for both.
12. Explain the various types of lists in HTML with example.
13. Explain the HTML table tag with example? How can you create irregular tables?
14. Show the use of HTML table tag to create Web page layouts. Give at least three examples.
15. Show the use of various attributes with HTML table tag.
16. Give an HTML file to create following table format.



17. How can you use images in Web pages? Give example.
18. What is HTML form? What is its purpose? Give the meaning of "action" and "method" attributes. Give a simple example.
19. List the various types of inputs for HTML form and discuss three of them with example.
20. Write an HTML form with two field sets: one for personal information and second for educational qualifications.
21. How do you organize directory structure of your Web site (i.e., structure of Web site)? What is its importance? Explain with suitable example.

Practice

Write an HTML document for each of the following using appropriate HTML tags to make them attractive and nicely formatted and test them in your browser.

1. Assume that you want to inform your friends to join a party. Send them an invitation showing all the details of the party.
2. Prepare your shopping list for the next weekends.
3. Prepare a Menu for the restaurant.
4. Prepare your last semester mark sheet.
5. Prepare a page of a magazine which you are reading regularly.
6. Prepare your single page resume.
7. Prepare a tutorial in a single Web page on any technical topic of your choice with table of contents on top of the page followed by contents. When any item is pressed from a table of contents, the pointer moves to that item within your tutorial.
8. Prepare a small personal Web site with 5 to 7 pages.
9. Prepare your Hall ticket for examination with your photograph.
10. Prepare a form to make your semester registration with your college.

Chapter Quiz

1. Try and find out the history of the various versions of HTML.
2. What are the four kinds of markup elements in HTML?
3. Use search engines to find out about the project Semantic Web. What is the role of markup language within it?

Useful Web Addresses

http://www.w3schools.com/html/
http://www.htmlgoodies.com/
http://www.w3.org/MarkUp/Guide/

CamScanner

4 From HTML to XHTML

LEARNING OBJECTIVES

At the end of this chapter, you will be able to:

- Understand the standardization processes for HTML, CSS and XML.
- Define XHTML and its types.
- Show the syntax of HTML 4 and XHTML.
- Give the document structure of XHTML and differentiate it from HTML.
- Discuss the use of the meta tags.
- List the character entities and show their use.
- Use frames and framesets to partition the contents of Web site.
- List and discuss the important features of HTML5.

Here we continue the exploration of HTML into XHTML. You will learn about the various standards that have been developed for HTML. More advanced HTML will also be studied, and the ability to control search engines, cache refresh and meta information.

This chapter charts the rise of XHTML (Extensible HyperText Markup Language) and important standards to correct the failings of early attempts at harmonization between different browsers. It also provides more advanced knowledge of HTML and XHTML:

1. The XHTML standard.
2. Syntax and document differences between HTML and XHTML.
3. The meta tag.
4. Web page caching, refresh, search engines and expiry.
5. Character encoding and entities.
6. Frames.

It finishes with a overview and features of HTML5 which is the latest update to HTML for future Web.

4.1 More History, More Standards

The last chapter introduced you to the practicalities of developing for the Web but before we go much further we need to take another brief look at history to understand where the Web is heading.

The Web was initially conceived as a place where resources were available to all. The idea was that information was linked together and not necessarily in a linear or even hierarchical fashion but that anything on the Web could be accessed from many different pathways and directions. It would not be a static source, like a book for example.

CamScanner

One of the main problems connected with this universality is that although the tools exist, in the form of HTML, to present the information, it is not experienced in the same way by everyone. There are a lot of factors that alter the perception of any resources placed on the Web such as the type of computer, the connection speed and the browser.

So when you develop pages and applications you must take into account these differences, which do alter the experience of the Web. The key here is to create pages so that the greatest number of visitors can view them as closely to the way you designed as possible.

In the past, efforts were made, in the famous 'Browser Wars', to segregate this experience more by creating extensions to HTML that only a specific browser could use. Netscape, in 1994, did just this and although people using their browser could see pages with colored text, images etc., those who didn't use it could not see things in the same way. The extensions created had some impact on the Web surfing community and many went over to using Netscape. Other companies, such as Microsoft, fought back with their own creation in the shape of Internet Explorer, which in turn had its own extensions that only it could recognize.

This continuing fragmentation into groups created problems for those designing pages. A large proportion of their time was taken up trying to develop ways to get around the proprietary tags that had been added to HTML. Multiple versions of the same Web sites had to be created, which had to be loaded according to which browser you were using.

Changes

All this confusion and demarcation of the Web led a lot of designers to believe that standards were the answer. The way this was and still is being achieved is through the World Wide Web Consortium (W3C), directed by the Web's inventor, Tim Berners-Lee. Universality was one of the main aims of the Web and it is this that the W3C tries to maintain while trying to keep pages looking good for everyone. The main contenders, Netscape and Microsoft, are now members, along with other companies that are involved, such as Adobe and Macromedia. The idea is to agree on a standard that everyone can use and that companies do not change.

The proprietary extensions that were developed were standardized while other specifications were removed from HTML. The browser developers were encouraged to support the new official specifications so that any pages written according to them would behave the same no matter what they were viewed on. This became HTML 3.2.

The next step was to develop this specification even further than before. The original idea behind HTML was to be able to create single pages in which content, structure and formatting instructions were present. The W3C thought a far better way of doing this would be to separate the formatting instructions from the content and structure, therefore giving the ability to apply them to the whole site rather than just one page. To do this a large number of formatting instructions were marked for removal in the next version of HTML. The new system for formatting was developed and called Cascading Style Sheets (CSS).

Initially CSS recreated the HTML aspects it was to replace, although slowly it became much better with further specifications. For example, it gained the capability to position elements on a Web page with far greater precision, enabling graphics to be displayed like old HTML but allowing professional layouts to be reproduced in the browser.

However, these improvements did not take away problems with the remaining proprietary extensions and HTML code that was badly written. Browsers did not help in this regard as they allowed a lot of mistakes to pass without any error messages being produced. In the process, this made the browser change the way a page would look. HTML needed some radical changes to become stable enough to develop further features.

The next step was XML (Extensible Markup Language).

XML

XML is a meta-language. It is a language used to describe and define other languages. In essence it looks a lot like HTML – it possesses tags, attributes and values – but you can use it to design your own custom markup language, which can then be used to format your own documents. Using XML it is possible, using tags, to identify data and this then makes the data available for use again. A program can then extract that information and manipulate it along with information from other sources and output the combination, possibly in an entirely different format.

However, XML is nowhere near as lenient as HTML and will not allow the mistakes and general sloppy pages that the latter does. The software that interprets the XML is known as a parser and demands that it is written in a strict way, paying attention to case-sensitivity, special punctuation and other details. The idea that the W3C went along with was to rewrite HTML in XML! This had a bonus in that any code written in the new specification could be understood by all the browsers already available. Another point was that anyone who already had some knowledge of HTML could quickly get used to it, provided they learnt the basic syntax required. Since it is built from XML it gains the same power and flexibility of that language too, providing a stable platform for CSS. The name chosen for this new form of HTML was XHTML.

While HTML and XHTML still co-exist, the likelihood is that XHTML is closer to what the future has in store. There are people who prefer one or the other and some who say that XHTML is the only way to go forward.

There are several types of XHTML, which allows some flexibility:

1. *Transitional*, which allows the use of deprecated (out of date, superseded) tags.
2. *Frameset*, which allows the use of both deprecated tags and frames.
3. *Strict*, which doesn't allow the use of any types of deprecated tag.

Each of these can be combined with CSS. A special subset of the XHTML 1.0 specification is also defined as the XHTML Basic 1.0 specification, which contains all the components of XHTML except problematic parts, like frames, which can cause problems for browsers on small devices.

It's advisable to avoid using deprecated tags if you can help it!

Checkpoint!

- Problems were initially caused in the development of HTML by a lack of standards.
- Browser makers tended to add proprietary extensions that limited those who could see the sites in the way that was intended.
- This has been termed the Browser Wars of 1990s.
- The W3C became the main source for standards that browsers were to follow.
- The evolution of HTML led to the separation of formatting instructions from content, leading to the development of CSS.
- HTML was redeveloped as XHTML, using XML, to produce a more strict approach to Web coding.
- XHTML provides a more stable platform for CSS.
- Several standards were produced: transitional, which allows the use of deprecated tags; frameset, which allows the use of deprecated tags and frames; and strict, which doesn't allow the use of deprecated tags.

CamScanner

4.2 The Move to XHTML

XHTML and HTML share a common vocabulary but have a slightly different syntax. Both XHTML and HTML 4 are demanding in the structure you imply on your code. For example, to start a transitional HTML 4 and an XHTML page we begin with a DOCTYPE declaration. This lets browsers and validators know how to judge code and check it appropriately.

For HTML 4:

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

This is followed by the usual <html> tag to begin the actual HTML part of the document.

For XHTML pages, because they are an XML application, a page should begin with an XML declaration:

```
<?xml version="1.0" ?>
```

The question marks say that this is an XML declaration. Be careful with spacing: there must be no space between the opening question mark and the xml, for example. The version should be also written like it is here – with quotes around it.

The XML declaration can also contain the character set encoding as an optional attribute:

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

Here the encoding format used is 8-bit Unicode Transformation Format. As you can see, any values being assigned the attribute must be contained in quotes. It's possible to specify foreign language character sets, such as those shown in Table 1.

For XHTML there are several possibilities as far as choice of document type goes, each using *Document Type Definitions (DTDs)*.

For XHTML 1.0 strict the tag would be:

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

Table 1 Available character sets

Encoding name	*Character set*
Shift_JIS	Japanese Kanji
ASMO-449	Arabic Farsi
greek7	Greek Cyrillic
KOI8-R	Russian Cyrillic
Big5	Chinese traditional characters

For XHTML 1.0 transitional it would be

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.00 Transitional//EN"
                "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
                transitional.dtd">
```

For XHTML-Basic 1.0 the tag is

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
                "http://www.w3.org/TR/xhtml1/DTD/xhtml-
                basic/xhtml-basic10.dtd">
```

For XHTML 1.0 frameset:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.00 Frameset//EN"
                "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
                frameset.dtd">
```

Like every document in XML, XHTML must have a root element that contains everything else. In this case it is the <html> element, which of course contains all your HTML between the opening and closing tags. The <html> tag may also contain some optional attributes.

One of the attributes is the actual human language being used in the form of a code, as shown in Table 2. It's used like this for English:

```
<html xml:lang="en">
</html>
```

The other important attribute is the declaration of its *namespace*:

```
<html xmlns="http://www.w3.org.org/1999/xhtml">
</html>
```

This is not a strict requirement as the DTD will supply it if it's not included.

Table 2 Human language codes

Language	*Code*	*Language*	*Code*
Hindi	hi	Korean	ko
Japanese	ja	Chinese	zh
Russian	ru	Norwegian	no
Swedish	sv	Danish	da
Turkish	tr	Greek	el
Portuguese	pt	Spanish	es
Italian	it	German	de
French	fr	English	en

CamScanner

Document Structure

The document structure for an XHTML document follows pretty much the format we have been using so far, with the additional parts above included:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.00 Transitional//EN"
                "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
                transitional.dtd">
<html xmlns="http://www.w3.org.org/1999/xhtml">
<head>
    <title>My first XHTML page</title>
</head>
<body>
    <p>What else can you say but... Hello World!</p>
</body>
</html>
```

The main difference here is that the head and body elements are required in XHTML whereas in HTML they are optional.

Some Other Differences

1. In XHTML it's important to remember to close tags in those elements that need it. HTML was relatively easy going and it was possible to get away without closing some elements properly. Some elements, such as the line break element `<br>` in XHTML, must contain a final forward slash: `<br/>`.
2. All XHTML tag and attribute names must be in lowercase and all attribute values must be enclosed in quotes.
3. You can nest tags but this must be done correctly without overlapping tags.
4. Attributes cannot be abbreviated and should appear fully written out.
5. An XHTML document must specify a document title.

The head section may again contain the language attribute as described earlier. It may also contain a profile attribute to be used to specify the URL for a meta data profile for the document.

Checkpoint!

- HTML and XHTML have similar languages but different syntaxes.
- XHTML is less forgiving.
- XHTML pages should begin with an XML version tag.
- Both HTML 4 and XHTML should have an appropriate DOCTYPE declaration.
- XHTML must contain a root element provided by the `<html>...</html>` element.
- XHTML may also have the language attribute attached to the `<html>` tag.
- A namespace can be provided for XHTML, although the DTD will supply it if it is absent.
- Both the head and body elements are required in an XHTML document.

Concept Check

- You can have any page you write checked by a validation service such as the W3C checker at `http://validator.w3.org/`. This will process any page that it is pointed at and provide a detailed report on any failings! Why not try running some of your new XHTML pages through it and see if you are up to scratch? It will also process and report on HTML 4 too.

4.3 Meta Tags

Metadata is information about information or, in this context more specifically, metadata is machine-understandable information about Web resources. It can be included in both HTML and XHTML to describe the actual document rather than the document's content. Metadata is included in the head section of your page:

```
<html>
<head>
  <title>Cave Dwelling</title>

  <meta name="author" content="Fred Flintstone"/>
  <meta name="keywords" content="stone age, lifestyles,
                 retro"/>
  <meta name="description" content="living prehistory">
  <meta name="robots" content="index,follow">

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

The meta tags are used to provide information to search engine robots; they pick up the data you give and also control to some extent where they go on your site following links. The sorts of information you can give include a general description of your site together with key words.

There is also an instruction here for search engines to index the page and follow any links. It is possible to stop the page being listed:

```
<meta name="robots" content="noindex"/>
```

Other possibilities are

```
<meta name="robots" content="index,follow">
<meta name="robots" content="noindex,follow">
<meta name="robots" content="index,nofollow">
<meta name="robots" content="noindex,nofollow">
```

Be sure not to give conflicting or repeating directives!

There is also a meta tag attribute called http-equiv that is used with a content attribute to create meta-functions. These functions can be used, for example, to give an expires date, refresh period or redirection.

In the head section to set an expiration date:

```
<meta name="expires" content="Mon, 20 Jul 2007 16:00:00 GMT"/>
```

CamScanner

This sets the expiry point when a page will be reloaded from the Web site, essentially to ensure a cache (a local store of frequently visited Web sites) is kept up to date. Putting a date of '0' will stop the page being cached at all.

Again to force a refresh after a period of time, in the head section:

```
<meta name="refresh" content="50;http://myownpages.co.uk/
                mynewssite.html"/>
```

This will cause the page to be refreshed after 50 seconds and a redirection to occur to the URL specified.

Memory Cache

As we have just seen, Web browsers can cache (store) pages for quick reviewing without having to request them again and re-download the document. Each page has a Time to Live (TTL), the time it is kept in the cache without going back and reloading it. This is usually 30 days, when the browser cache has been cleared or the allotted memory is all used up.

A browser can be stopped from caching a page, if it supports the `<meta>` element's `http-equiv` attribute. This forces the browser to ignore the cached page and instead make a request again. To do this the value `pragma` is assigned to the `http-equiv` attribute and a `no-cache` value to the `content` attribute:

```
<html>
<head>
     <title>no cache example</title>

     <meta http-equiv="pragma" content="no-cache"/>
</head>
</html>
```

A search engine can also cache your page and offer it as an alternative if your site is down or inaccessible for some reason. The only problem here is that it may be very out of date! You can stop a search engine from archiving a page by using

```
<meta name="robots" content="noarchive"/>
```

Again, this should be placed in the head section.

Formatting with Scheme

Another useful meta element attribute is named `scheme`. This can be used to give a context for data such as date and time, which can use several formats:

```
//Here scheme="USA" implies "MM-DD-YYYY"
<meta scheme="USA" name="date" content="09-21-1966">

//Here scheme="Europe" implies "DD-MM-YYYY"
<meta scheme="Europe" name="date" content="21-09-1966">
```

Concept Check

- Write a new Web page (or alter an old one) to incorporate some metadata such as author, description, key words.
- Make the page so it isn't cached by the browser.
- How do we also stop it being cached by the search engine?

4.4 Character Entities

What happens if you want to tell someone about HTML on your Web site? Could you just write the following?

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
        "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
     <title>tell them about HTML</title>
</head>
<body>

This is how you set up a link:

<a href="www.google.com">Google!</a>

</body>
</html>
```

No, because what you would get is the link to the place itself! How would you get the writing of `href` etc.? The answer is to use character entities to describe what you want printing, then it won't be processed by the browser as HTML. For example, the < (less than) symbol can be replaced by the character entity `<` while the > (greater than) symbol can be replaced by the character entity `>` so, it's possible to replace a great many of the special HTML characters with this special set of entities and therefore display HTML code itself:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
        "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
     <title>tell them about HTML</title>
</head>
<body>

This is how you set up a link:

<a href="www.google.com">Google!</a><br/>

&lt;a href=&quot;www.google.com&quot;>Google!</a>

</body>
</html>
```

Character entities can be useful for inserting symbols that aren't usually available or just for placing spaces around text in your code:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
        "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

CamScanner

```
<head>
    <title>tell them about HTML</title>
</head>
<body>

Less than &lt;<br/>
Greater than &gt;<br/>
Ampersand &amp;<br/>
Divide &divide;<br/>
Copyright &copy;<br/>
Registered &reg;<br/>
Quote &quot;<br/><br/>

    over here!

</body>
</html>
```

The complete list of entities is very large. Table 3 contains some of the most useful.

Checkpoint!

- Metadata provides information about the document not the content.
- The `<meta>` tag can provide information by these:
  - keywords – a list of keywords for search engines
  - description – a description of the pages content
  - robots – information for search engine crawlers
  - author – who is the author?
  - expires – the date the page expires in the cache
  - refresh – force a refresh after a specified time
  - `http-equiv` – stop caching using `pragma` value
  - scheme – give a local context for items such as dates.
- Character entities provide a way of outputting special characters within HTML pages.

Concept Check

- Write a Web page that displays a valid HTML document, including a hyperlink and a list of five items.
- How do you output the trademark and copyright symbols?

Table 3 Useful character entities

Character	*Character Entity*	*Description*
		Space
&	&	Ampersand
"	"	Quote
<	<	Less than
>	>	Greater than

(Continued)

Table 3 (Continued)

Character	*Character Entity*	*Description*
©	©	Copyright
®	®	Registered
™	™	Trademark
£	£	Pound
¢	¢	Cent
÷	÷	Divide

4.5 Frames and Framesets

A way of dividing content on your Web site when it starts to get large is to use frames to break information up and partition it. For example, you could place a separate frame down one side of the screen and use it to place navigation aids and links while the main content is in a separate frame to one side.

Rows

To separate the different elements of your content it is possible, for example, to divide the page into rows and to place a new page into each frame:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Rows of Frames</title>
</head>
<frameset rows="220,*,60">
<frame name="banner" src="banner.html" scrolling="no"/>
<frame name="main" src="mainpage.html"/>
<frame name="options" src="options.html" scrolling="no"/>
</frameset>
</html>
```

The other pages are simply normal XHTML pages, which are individual files. The first one is `options.html`:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>options.html</title>
</head>
<button type="submit" id="abutton" name="abutton"
            value="mybutton1">button 1</button>
```

CamScanner

```
<button type="submit" id="bbutton" name="bbutton"
            value="mybutton2">button 2</button>
<button type="submit" id="cbutton" name="cbutton"
            value="mybutton3">button 3</button>

</html>
```

The next is banner.html:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>banner.html</title>
</head>
<img src="banner.jpg">
</html>
```

The last one is mainpage.html:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>mainpage.html</title>
</head>
<body>
This holds the main content!
</body>
</html>
```

The output for this HTML code is shown in Fig. 1. The main parts to the HTML that describes the frames are the <frameset>, <frame> and closing tag </frameset>. Remember here in XHTML you also have to use the appropriate <!DOCTYPE> for framesets. The frameset tag in this case contains an attribute, rows, which declares the divisions either in terms of pixel or percentage size. You can, for instance, say you want rows that are two rows of 100 pixels wide with the remainder used for another element of the content by using the asterisk. It is also possible to use more than one asterisk at a time, which will divide the remaining space equally between them. To divide space unequally among such content, add a number to the asterisk, such as 2*, in which case between two, two thirds would go to one and the last third to the other. Another way to describe the frames is by percentages of the area available, using the number followed by the percentage sign.

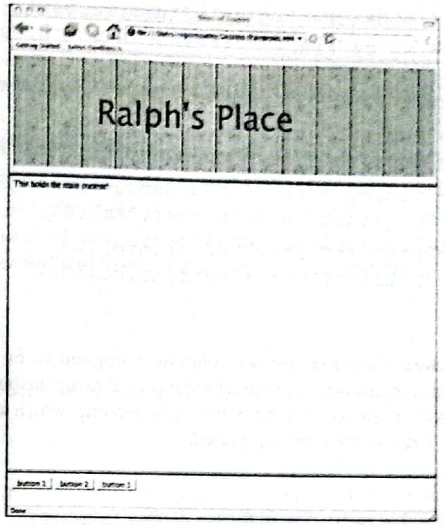


Figure 1 Basic frames in rows.

The name you give in the <frame> tag allows you to identify it later and target more content at it.

Alternative Content

Note that you only give a body section to such a program where you want to support browsers that don't have frame capability. Alternative content is given by adding:

```
<noframes> <body>
Alternative content, for those without frame capability.
</body></noframes>
```

Columns

To create content that goes across the page in columns rather than in rows, simply change the attribute to cols rather than rows and use the numbers in a similar way:

```
<frameset cols="100,150,*">
```

The columns are divided from left to right so the left-most one above would be 100 pixels, followed by 150 pixels with the remaining area being used for the last. Again the area can be specified as a percentage.

Frames Using Columns and Rows

Frames can divide a page by rows *and* columns too. For example:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

CamScanner

```
<head>
    <title>Rows of Frames</title>
</head>
<frameset frameborder="3" rows="*,*,*" cols="*,*">
<frame name="topleft" src="banner.html"/>
<frame name="topright" src="openingpage.html"/>
<frame name="midleft" src="openingpage.html" scrolling="no"/>
<frame name="midright" src="banner.html"/>
<frame name="botleft" src="banner.html"/>
<frame name="botright" src="buttons.html"/>
</frameset>
</html>
```

This creates a series of frames three rows by two columns, filling out to fit the area available, using the asterisk. Notice in this example, the attribute frameborder is used to set the border width – this can also be set to 0 for no border. Another attribute for the frame tag is scrolling, which allows the frame to be moved with an added bar. Figure 2 shows the frames displayed.

Nesting Frames

Frames can also be nested within each other, to further divide a frame's space:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
```

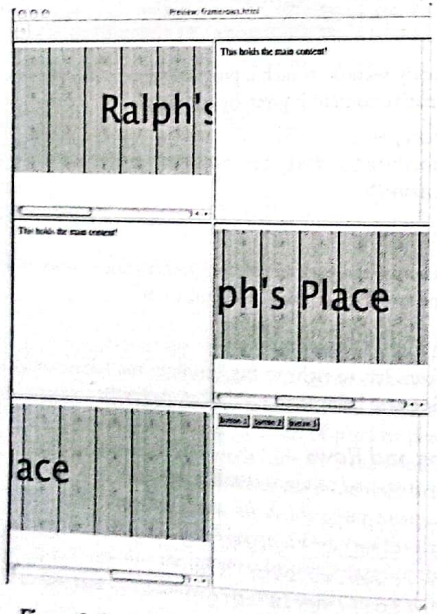


Figure 2 Basic frames using columns and rows.


```
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Rows of Frames</title>
</head>
<frameset cols="20%,80%">
    <frame name="main" src="buttons.html">
<frameset rows="25%,75%">
    <frame name="top" src="banner.html" scrolling="no"/>
    <frame name="bottom" src="openingpage.html"/>
</frameset>
</frameset>
</html>
```

Figure 3 shows how this HTML code divides the screen into useful areas for navigation, main content and banner.

Inline Frames

Content can also be mixed in another way and placed with other graphics and text. To do this, an inline frame can be used:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

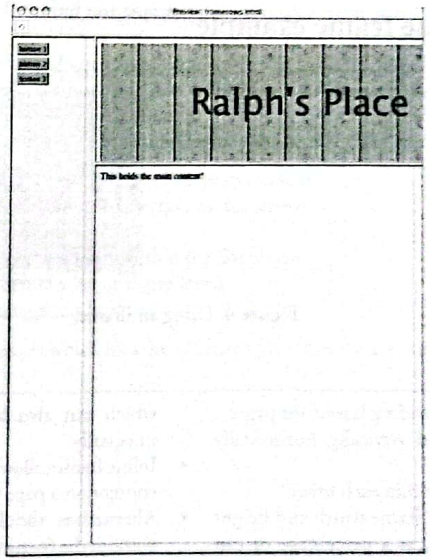


Figure 3 Nesting frames.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Rows of Frames</title>
</head>
<h1>An inline frame example</h1>
Inline frames can be used to sit with other types of content
                        such as text

and graphics.
    <iframe src="banner.html" name="mybanner" width="300"
                height="200" align="right">
This text will appear if the browser can't display the frame
</iframe>
The iframe behaves in some ways like an image in that text will
                tend to flow around it.
Be careful using align as it is deprecated.
</html>
```

This example is shown in Fig. 4.

The example actually uses a *deprecated* attribute `align`, which means it has been superseded by some other way of achieving the same effect. The `align` attribute can also be used with `image` for the same effect, forcing text to flow around it. The CSS `float` property has now replaced this capability.

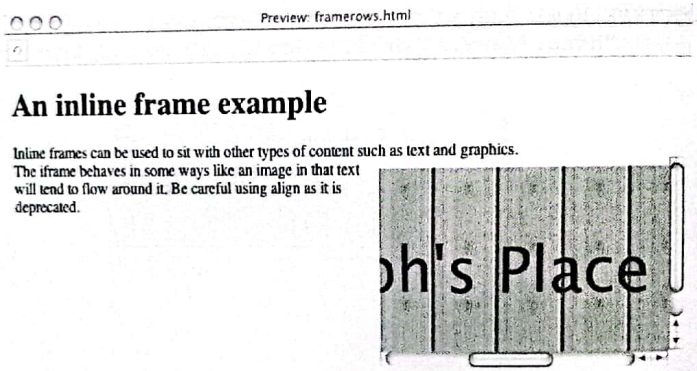


Figure 4 Using an iframe.

Checkpoint!

- Frames are a way of providing layout for pages.
- Frames can be arranged vertically, horizontally or a mix of the two.
- Frames can be nested within each other.
- The parameters for the frame width and height can be stated in pixels, as a percentage or can use the rest of available space using asterisk *, which can also be used to divide the space unequally.
- Inline frames allow frames to be mixed with other content on a page such as graphics and text.
- Alternatives should be provided for browsers without the frame capability. The `<noframes>` element can be used for this.

Concept Check

- Draw out a rough design for a Web page using frames.
- Try and incorporate graphics and text that are controlled by a frame layout.
- Implement the design and experiment with the layout; can you get it exactly as you had designed it?
- Now try and embed an inline frame on a page with graphics and text.

4.6 HTML5: Overview and Features

We have studied the various features of the HTML and XHTML in previous and current chapters, respectively. The HTML has undergone different versions with improved and new features added in each of them. The HTML4 (also HTML 4.01) is widely used version for Web development as it included many important features. However, the advancement in Web technology and demands from modern Web applications specifically the Web 2.0 applications required more features that are not supported by the browsers implementing HTML 4.0. To support such features, the browsers use plug-ins. For example, to incorporate multimedia features, one needs to use FLASH. To overcome all these limitations, the new version called HTML5 is released in 2009 with many features. Let us take the overview of the new features of the HTML5 with some of them in details for getting the real feel of the HTML5.

The HTML5 is a major revision and considered as next generation Web development which supports the Web development with desktop like behavior. The meaning is that the user while using a Web application gets same experience as the desktop applications. Another major advantage of using HTML5 is that it incorporates the features in form of elements or attributes for modern applications, and users need not use additional software in form of plug-ins. For example, the AUDIO and VIDEO are elements and the browser supporting HTML5 need not require plug-in to play them as it becomes integral part of the browser itself.

The HTML5 adds many new elements and new attributes to the elements. These new elements and attributes are provided not only for improved user experience but also important from the SEO (Search Engine Optimization) point of view. For example, the following new elements are very useful to group the contents as per the need so that using them search engine can identify them and rank the page accordingly.

`<header>` – used to define a header for section or document
`<footer>` – used to define a footer for section or document
`<article>` – used to define an article
`<section>` – used to define a section within the document
`<menuitem>` – used to define a list of menu items
`<nav>` – used to define navigation links

Consider the following example which uses the `<header>`, `<footer>`, `<menu>` and `<menuitem>` elements.

```
<!DOCTYPE html>
<html>
<body>

<header>
    <h1>My Header</h1>
<header>
```

CamScanner

```
<div style="background:yellow;padding:10px;"contextmenu="mymenu">
    <p>Right-click here </p>

    <menu type="context" id="mymenu">
        <menuitem label="Hello"></menuitem>
        <menu label="Share your views on ...">
                <menuitem label="Facebook" ></menuitem>

        </menu>
            <menuitem label="Send Mail" onclick="window.
location='mailto:?body='+window.location.href;"></menuitem>
        </menu>

</div>

<p>Use Firefox </p>

<footer>
    <h1>My Footer</h1>
</footer>

</body>
</html>
```

As can be seen in the above example, the document starts with the header, followed by the division section defined with the element containing a menu with three menu items namely "Hello", "Share

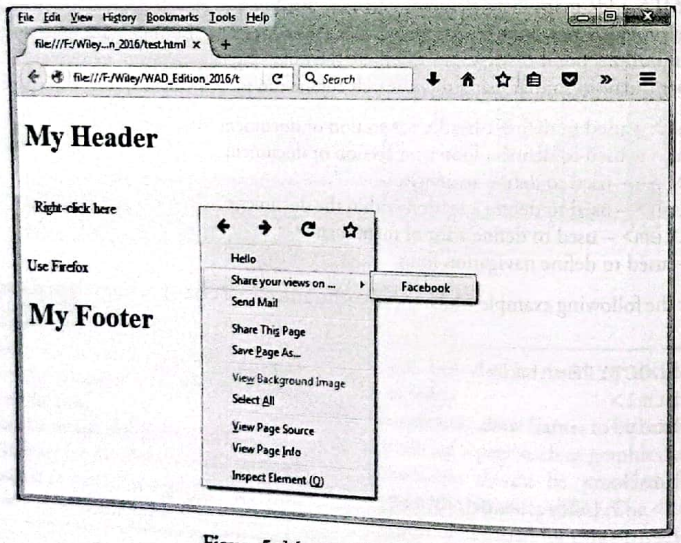


Figure 5 Menu with HTML5.

your views on ..." and "Send Mail" (Fig. 5). The second menu item contains submenu with one menu item "Facebook". With each menu item, you can also add the event that points to a function to be executed when that event occurs on the menu item. The third menu item "Send Mail" uses event "onclick" which causes "outlook" to start when it is clicked. The output of the above example is shown in Fig. 5.

When the right click mouse button is clicked on yellow part, the menu is shown as in Fig. 5. This provides the feeling of using desktop application even while interacting with Web page. The menu can be made rich in terms of functionality by adding different events to it.

Another important addition in the HTML5 is multimedia capabilities. Often, Web pages contain the multimedia content like pictures, audio, video, animation, etc. The browsers support them using the plug-ins, i.e. additional software. With the <audio> and <video> elements supported by HTML5, it becomes history. The audio and video contents are available in different formats. The HTML5 supports MP3, WAV and Ogg audio formats while for video it supports MP4, WebM and Ogg formats. Consider the following HTML code using the <video> element of HTML5.

```
<!DOCTYPE html>
<html>
<body>

<video width="400" height="300" controls>
        <source src="myvideo.ogg" type="video/ogg">
        No browser support for HTML5 video.
</video>

</body>
</html>
```

As can be seen, the attribute `controls` is used to show the video controls like play, pause and volume. It is necessary to use the width and height attributes, otherwise it causes the problems in Web page. The <source> element is used to provide the video file to be played. If we want to play the video automatically, we can provide attribute `autoplay` as

```
<video width="400" height="300" autoplay>
  ...
<video>
```

Following is an example of the <audio> element in HTML5.

```
<!DOCTYPE html>
<html>
<body>

<audio controls>
        <source src="myaudio.mp3" type="audio/mpeg">
        No browser support for audio element.
</audio>

</body>
</html>
```

The output of the above code is shown in Fig. 6. The use of attribute `controls` shows the various audio controls as shown in Fig. 6.

CamScanner

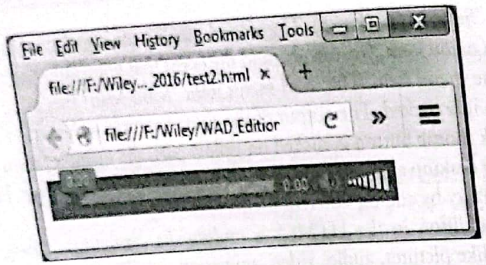


Figure 6 Audio in HTML5.

Note that different browsers have different support for HTML5 audio and video formats. Hence, before using the audio and video formats supported by HTML5, check for the browser compatibility.

The HTML5 supports the `<canvas>` element to allow the user to draw graphics. The canvas acts as a container and graphics is drawn using the JavaScript. The syntax for the `<canvas>` element is as follows.

```
<canvas id="mycanvas" width="200"
height="200"></canvas>
```

It is necessary to provide id to the `<canvas>` element as we need to refer to it in JavaScript to draw the graphics in the canvas area defined using width and height. The following example draws a square with red color in the middle of the canvas area.

```
<!DOCTYPE html>
<html>
<body>

<canvas id="draw_area" width="200" height="200"
style="border:2px dotted #000000;">
</canvas>

<script>
        var can1 = document.getElementById("draw_area");
        var ctx = can1.getContext("2d");
        ctx.fillStyle = "#FF0000";
        ctx.fillRect(50,50,100,100);
</script>

</body>
</html>
```

The above example first defines the canvas with area 200 × 200 with an id `draw_area`. Then the JavaScript code is used to draw a square of size 100 × 100 in the middle of the canvas area surrounded by dotted border. The `document.getElementById("draw_area")` function returns the reference to our canvas and named as canvas object `can1`. Then the `can1.getContext("2d")` function returns the 2D graphics context object whose methods are used to draw the required graphics. The output of the above code is shown in Fig. 7. The HTML5 `<canvas>` element supports rich graphics with the different graphic functions.

Apart from the above features, the HTML5 supports many more features. Import features are listed below.

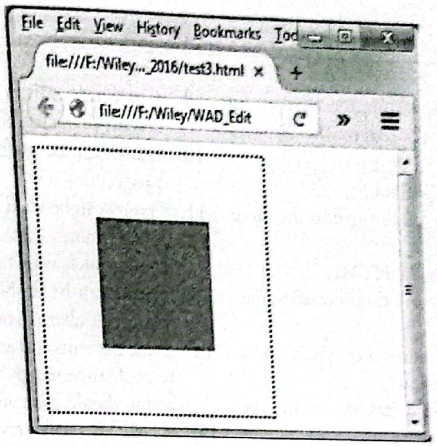


Figure 7 Graphics using canvas.

- **Web Forms 2.0:** It is supported with new attributes for `<form>` and `<input>` elements as well as added values for type attribute.
- **Client Side Storage:** It allows users to store the data with three options, namely session, local and database. The session data is removed once the browser session ends. The local data is persistent data storage on user computer. The database storage allows data stored locally using SQL on user machine.
- **Web Sockets:** They allow the communication between client and server.
- **Web Workers:** They allow JavaScript running in background without affecting the user interface scripts.
- **Drag and Drop API:** They allow the elements to be dragged and dropped on the Web page.
- **Geolocation:** It is used to find your geographical location and share it.

Checkpoint!

We now know that:

- HTML5 is a latest version of HTML supporting desktop-like behavior for Web applications.
- HTML5 adds many new elements and attributes to encourage the development of rich Internet application.
- HTML5 supports multimedia in-built using audio and video elements.
- HTML5 provides graphics support using canvas.
- HTML5 also supports Web Forms 2.0, Client Side Storage, Web Sockets, Web Workers, Drag and Drop API and many more.

Concept Check

- Explore the 2D graphics with HTML5 to draw real drawings.
- Explore the use of Web Forms 2.0 to design the forms with advance features.
- Research on support available for various HTML5 features in leading browsers.

CamScanner

Summary

1. The initial struggles within the market for dominance of browsers led to the Browser Wars in the 1990s.
2. Standards are very important.
3. HTML evolved from 3.2 through to the more strict XHTML.
4. XML was used to 'rebuild' HTML.
5. XHTML has several levels: transitional, frameset and strict.
6. XHTML demands a more strict syntax and a more structured document.
7. Metadata provides information about the document, not content.
8. Metadata can direct how a Web page is dealt with by browsers, servers and search engines.
9. Character entities can be used to encode certain characters that HTML uses.
10. Frames can be useful to break up the layout of a page.
11. Frames can be sized and structured as required.
12. Inline frames can be mixed in with other content such as text and graphics.
13. Frames can be problematic for some browsers so provide alternative content where possible.
14. Some elements and attributes have become deprecated, superseded by new language features and so should generally be avoided.
15. HTML5 is a major revision of the HTML which allows users to develop web applications with desktop-like behavior. It includes many new features enabling next generation of web development.

Key Words and Phrases

Browser Wars A name given to the competition between Web browsers for dominance in the marketplace, particularly in the late 1990s.

Cache A memory store that can hold frequently requested Web pages.

Character entity Certain characters can be encoded as HTML entities, which can take their place. For example '<' can be encoded as `<`

CSS Cascading Style Sheets, a language used to describe the presentation of a structured document in HTML and XHTML.

Deprecated A feature of software or a programing language is said to be deprecated when it is phased out or made obsolete.

Driver A software component that provides an interface between different levels in a computer system, usually between software and hardware.

DTD Document Type Definition, a declaration in XML that specifies constraints on a document, for example where it is valid XHTML.

Frame A feature of HTML that allows layout to be broken into separate areas.

HTML5 It is a major revision of HTML which includes many new features allowing user to develop Web applications with desktop-like behavior.

Inline frame A frame that can be set out with other text and graphics.

Meta tag A meta tag gives information about the page rather than its content.

Operating system The system software responsible for management and interaction between applications and hardware.

SEO Search Engine Optimization, process of improving the visibility of Web site or Web page on search engine results.

TTL Time To Live, the length of time before a page expires.

XHTML Extensible Hyper Text Markup Language has the same capabilities as HTML in terms of expression but is much more strict in syntax.

XML Extensible Markup Language, a general purpose markup language.

Multiple-Choice Questions

1. XHTML stands for
   (a) Extensible HyperText Markup Language
   (b) Extended Hyper Text Markup Language
   (c) Extension Hyper Text Markup Language
   (d) None of the above
2. What is major difference between HTML and XHTML?
   (a) `<head>` and `<body>` tags are compulsory in XHTML
   (b) Tags and attributes must be in lowercase with attribute values in double quotes in XHTML
   (c) Document title must be specified in XHTML
   (d) All of the above
3. Which of the following is correct in XHTML?
   (a) `height=200`
   (b) `height="200"`
   (c) `HEIGHT=200`
   (d) `HEIGHT="200"`
4. Which of the following elements is not mandatory for XHTML document?
   (a) doctype
   (b) title
   (c) body
   (d) All are mandatory
5. What is true about meta tag?
   (a) Used to provide information to search engines
   (b) Part of `<head>` tag
   (c) Describe the document
   (d) All of the above
6. Which of the doctype is common for XHTML?
   (a) Strict
   (b) Transitional
   (c) Frameset
   (d) All of the above
7. Which of the followings prevents the Web page being cached?
   (a) `<meta name="expires" content="0" />`
   (b) `<meta http-equiv="pragma" content="no-cache" />`
   (c) Both of the above
   (d) None of the above
8. Which of the following tags will not be used when frameset is used?
   (a) `<head>`
   (b) `<body>`
   (c) `<title>`
   (d) None of the above
9. If your browser does not support frames, which of the following tags is used to provide alternative content?
   (a) `<alt> .. </alt>`
   (b) `<noframes> .. </noframes>`
   (c) `<frameless> .. </frameless>`
   (d) None of the above

Review Questions

1. What were problems with HTML due to browser issues? How does standardization solve these issues? Name the standards developed by W3C to resolve them.
2. What is relation between XML, HTML and XHTML? What are the different types of XHTML?
3. What is DOCTYPE declaration? What is its importance? Explain with example in case of both HTML 4 and XHTML.
4. Give and explain the document structure for XHTML.
5. Differentiate between HTML and XHTML.
6. What do you understand by meta tags? List the commonly used meta tags with their purpose.
7. How do you stop page being cached by browser and search engine using meta tag? Explain with syntax.
8. What are character entities? Show their use with example.

CamScanner

9. What is frameset? Show its use in designing a Web site.
10. List the new features of HTML5 and explain important features with example.
11. Discuss the support of audio, video and graphics in HTML5.

Practice

Write the Web pages for the following strictly following appropriate XHTML type.

1. Prepare an application for applying for the post of software engineer in IT Company.
2. Prepare a banner for the technical event to be organized in your institute.
3. Prepare an information brochure for the one week course on "Web Technology" to be organized by your department.
4. Using frameset, design a Web site for the techno-event to be organized by your institute giving all the information.
5. Learn the various features of HTML5 using various Web resources or online tutorials.

Chapter Quiz

1. Find out a bit more about the origins of XML: which language is it a subset of?
2. What other languages have been developed from XML?
3. Why do some people recommend *not* using frames?
4. Explore, how does HTML5 provide desktop-like behavior to Web applications.

Useful Web Addresses

http://www.w3schools.com/xhtml/
http://www.w3.org/TR/xhtml1/
http://www.xhtml.org/
http://www.w3schools.com/html

Answers

1. (a)
2. (d)
3. (b)
4. (d)
5. (d)
6. (b)
7. (c)
8. (b)
9. (b)

5 Getting Some Style: CSS

LEARNING OBJECTIVES

At the end of this chapter, you will be able to:

- Define Cascading Style Sheets (CSS) and understand its purpose.
- Describe the basic syntax and structure of CSS.
- Use class, id and pseudo-class selectors for making styles more attractive.
- Describe, write and use internal, external and inline CSS.
- Apply styles to background, text, fonts and other HTML components.
- Define borders, margins and padding using styles for various HTML elements.
- Use absolute, relative or z-index method for positioning the elements.
- Handle the floating elements.
- List and use the important features of CSS3.

In this chapter you will learn how to present and control the format of Web pages using CSS. This includes the ability to precisely control the positioning and attributes of content while maintaining the structure of the document itself.

This chapter explains why we need Cascading Style Sheets (CSS) and how it enables the designer to separate presentation and formatting from content. It looks at the simple structure and syntax used and the basic method that enables the styles created to be linked to pages.

CSS allows you to enhance your Web pages in a way that HTML on its own never could and to keep a logically structured document in the process. Here it is shown how to add background images and colors and to manipulate text and other elements. It is possible to add borders, boxes and alter margins.

In HTML, positioning and layout either occur during the natural flow of the page or by using tables to control the positioning of elements. In CSS, precise control is enabled over where objects are placed either relatively or finely, with absolute pixel precision! At the end, features of CSS3 are discussed which are used to make Web very attractive.

5.1 The Need for CSS

In the early 1990s HTML was largely still concerned with the structural elements of a document. These elements included descriptions of paragraphs, lists, headings and hyperlinks. There was nothing to control layout in the form of tables or frames, for example. The main aim at this point was to have a language that cleanly described the structure rather than the presentation.

Over time, as more browsers such as Mosaic joined the WWW, additions were made to the language to provide features that included more control over displayed text and graphics. There was a demand with the surge of interest that features such as bold and italics control should be added. Marking up a piece of text to

CamScanner

be emphasized could mean that it was either made bold or placed in italics, and browsers could determine it one way or the other. What the author created was not always how the reader saw it in the end.

HTML (both 3.2 and 4.0) had large sections given over to presentation aspects such as color, size of text and background. Text was also made to be able to blink through HTML. The amount of HTML in a document given over to this kind of graphical presentation eroded the use of HTML as a structural markup language.

Structural meaning became lost and documents could not be processed in any way beyond providing output in a browser. For example, a speech processor could not rely on the information contained in a document to give it clues about sections or emphasis and there would be no strict rules for early HTML in any case!

If a page has no structure then it becomes difficult to use in several ways:

1. Like any poorly structured program in software engineering it makes it hard to follow and therefore difficult to understand what's actually happening. This makes it difficult to update and maintain.
2. Search engine indexing becomes more difficult as the lack of structural markup gives a lack of importance to key content.
3. Processing by software other than a browser is problematic as there are few hints at specific information.

It soon became obvious, especially to the W3C that this mixing of presentational and structural information could not continue and they began to work on a solution. The basis of this, revealed in 1995, was CSS. This was quickly taken on board and by 1996 had become a full recommendation on a par with HTML.

The initial CSS has been supplemented by a second level, CSS2, which became a full recommendation in 1998. This builds on CSS1, adding extensions without major changes to the actual first specification. CSS3 is the latest update which is developed as modules.

5.2 Introduction to CSS

CSS has improved the original messy attempt at controlling presentation within HTML:

1. Control of text color of any element.
2. Control of background colors.
3. Control of borders around elements.
4. Spacing between elements and borders.
5. Text manipulation and decoration.

Everything that was possible at the height of the HTML presentation days is possible in CSS and more. To give a basic idea of how CSS works and how powerful it is, a simple example is required.

A heading could be given in markup as:

```
<h1>The main heading</h1>
```

If you wanted to add color, emphasis and font control to this using the old way within HTML, there would be a lot of complex layout information to add. With CSS it's possible to do this quite simply by a selector, which in this case is `h1`, followed by a list of properties and values:

```
h1 {color: red; font: italic 1em Times, serif; text-decoration:
underline; background: black;}
```

If you wanted to change every `h1` tag to a particular set of colors and attributes the old way in HTML, you would have to go through your code and do this individually. CSS allows a particular style to be attached to a selector, which is usually an HTML element.

5.3 Basic Syntax and Structure

A CSS document, or section, consists of a list of rules containing a selector and declaration block.

Rules

The basic idea behind the syntax of CSS is simply:

```
selector {property:value; ....}
```

The selector is the identifier of the element, followed by a list of paired `property:values` enclosed within curly brackets. For example:

```
body {color: yellow}
```

Values that have multiple words must be enclosed within quotes:

```
p {font-family: "sans serif"}
```

If there is more than one property to be set then they are separated by semicolons:

```
p {text-align:center;color:red}
```

It's possible to break these up on to different lines rather than placing them all on one line:

```
p {
text-align:center;
color:red
}
```

which is useful if you have a lot of properties, as it makes them more readable.

As well as attaching quite a few properties to one selector, it is possible to have many selectors. So, to apply a set to `h1` and `h2`:

```
h1, h2 { color: red; font: italic 1em Times, serif;
                text-decoration: underline; background: black; }
```

or to make a whole group of headers red:

```
h1,h2,h3,h4,h5,h6
{
color: red
}
```

Classes

It's possible that sometimes you will need to make several styles for the same HTML element:

```
p.right {text-align: right}
p.left {text-align: left}
```

CamScanner

To actually use this within your HTML document:

```
<p class="right">
This paragraph will be right-aligned.
</p>
<p class="left">
This paragraph will be left-aligned.
</p>
```

You cannot specify more than one class for an HTML element though! A useful technique is to specify a class without its main tag so it can be applied to any other class that the designer wants to use it for:

```
.left { text-align: left }
```

You can then apply this to multiple tags as long as they share the same class:

```
<h1 class="left">
This heading will be left-aligned
</h1>
<p class="left">
This paragraph will also be left-aligned.
</p>
```

ID

Similarly, there is the `id` selector, which will apply a specific style to an identified element. For example, to declare a style rule with an `id`:

```
p#bluepara
{
text-align: center;
color: red
}
```

This will create a style rule that will match a paragraph with the `id` value `bluepara`. It will match for example

```
<p id="bluepara">Some interesting thoughts...</p>
```

but not

```
<h1 id="bluepara">This section...more thoughts</h1>
```

You could match any element with the `id` of `bluepara` by using

```
#bluepara
{
text-align: center;
color: red
}
```

The difference between `class` and `id`, although similar, is that `class` may apply to several parts of a page whereas `id` applies only to one and it should be *unique*. For example, it would be wrong to do this with the same `id`:


```
<h2 id="alert">This is an alert!</h2>
<p id="alert">This is an alert!</p>
<h3 id="alert">This is an alert!</h3>
```

Pseudo-Class Selectors

Some selectors can be considered different because of the way the element they belong to works. For example, the anchor that creates a link between documents can have pseudo classes attached to it simply because it is not known at the time of writing the markup what the state will be. It could be visited, not visited or in the process of being selected (clicked upon, or active). To catch these states and say what we want to happen in each, we can use pseudo-class selectors:

```
a:link {color: red}
a:active {color: yellow}
a:visited {color: green}
```

So, a link in this instance would be initially red; when visited it would be green; and in the process of being clicked on it would be yellow. Note here, in the case of pseudo classes, the use of the colon (:) in the rule rather than a dot (.) separator, as used in the usual classes.

There are more pseudo classes; for example

```
a:hover {font-weight: bold}
```

This will make a link bold when the cursor is above it. However, `hover` is not constrained to links so it may be possible to have other elements using the same `hover`. To make it specific to links use

```
a:link:hover {font-weight:bold}
```

Checkpoint!

We now know:

- the main idea behind CSS is to separate presentation from content.
- CSS controls lots of formatting and layout that used to be part of HTML.
- this method helps structure the document and is block oriented.
- a CSS document, or embedded section, consists of a list of rules.
- elements, or blocks, can be marked up with a name for classes or identifier tags.
- hyperlinks can also have formatting applied to them.

5.4 Using CSS

To use a style sheet, you can either embed the rules in the document you want to apply it to or have a totally separate file.

External Style Sheets

An external file is a good idea when you have a number of pages, or even a complete site, which you need to control in terms of presentation. Obviously this saves a lot of effort as, at one time, you would have needed to alter each page individually. This way you can change the look of the entire site by altering only one file. Each page that uses the style sheet needs to have a link to associate it with the style document:

CamScanner

```
<head>
<link rel="stylesheet" type="text/css"
href="mystyle.css" />
</head>
```

The work is done with the link tag, which needs to be inside the head section.

The actual style document contains lists of rules to apply and has the extension .css. The file can be written with a normal text editor as it contains just plain text. It should not have any HTML elements in it.

Listing 1 is an example of an external CSS file. It shows seven CSS rules with multiple properties. Note here the correct syntax of a comment, which is useful for particularly large files where you can annotate the styles as you wish.

```
body {
    background-attachment: fixed;
    background-repeat: repeat;
    background-image: url(images/test.gif);
}
font {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 12px;
    color: #0000FF;
}
.bodytext {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 16px;
    color: #000000;
}
.pageH {
    font-weight: bold;
}
.bluesubheading {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 14px;
    color: #0000FF;
    text-decoration: underline;
    font-weight: bold;
}
.bluebodytext {
    font-family: Arial, Helvetica, sans-serif;
    color: #0000FF;
}
.subheading {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 18px;
    font-weight: normal;
    text-decoration: underline;
}
```

Listing 1 External CSS file.

Embedded Style Sheets

It is also possible to use an internal style sheet; that is style information that is attached to only one document. Here the CSS information is contained within the head section, for example

```
<head>
<style type="text/css">
hr {color: sienna}
p {margin-left: 20px}
body {background-image: url("images/back40.gif")}
</style>
</head>
```

Any information is given using the style tag followed by the formatting data. One problem here is that when an older browser loads the page, it will ignore the <style> tag then print the CSS information following it on to the browser page! To get around this the best way, enclose any such code within an HTML comment element:

```
<head>
<style type="text/css">
<!--
hr {color: sienna}
p {margin-left: 20px}
body {background-image: url("images/back40.gif")}
-->
</style>
</head>
```

New browsers that are processing the CSS information will ignore the comments but older ones will see the comments and not process the information between the start and end of the <style> tag.

Another way of incorporating CSS formatting is to use an inline method; that is, to write the markup into the actual element that requires it:

```
<p style="color: sienna; margin-left: 20px">
This is a paragraph
</p>
```

Although this can be done it does lose some of the advantages of CSS by mixing the content and presentation. It is probably best used when there is a unique occurrence of a style in a site or page.

It's also possible to have several active style schemes, for example an external style sheet and an internal style sheet. In this case the more specific values will be inherited from the internal style sheet.

5.5 Background Images, Colors and Properties

Using CSS it is possible to control the color of an element's background, use an image in the background, tile an image (repeat it horizontally or vertically) or place an image at a specific place on a page.

CamScanner

Background Color

To change the background color to yellow, you might place the following in your external CSS file:

```
body {
    background: #F0F8FF;
}
```

or

```
body {
    background: yellow;
}
```

Remember colors can be set using RGB color codes (hex or decimal) or names. The W3C CSS standard supports only 16 names – aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white and yellow, although most browsers support more (these can be found at `http://www.w3schools.com/css/css_colornames.asp`).

To make sure any text is black you can use:

```
body {
    background: #FFFF20;
    color: #000000;
}
```

To test these out you need an HTML page that links them in:

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Testing CSS</title>
    <link rel="stylesheet" type/css" href="mycss.css"
                title="green">
</head><body>
How about some color...
</body></html>
```

Notice here that the background property sets all background properties that include `background-color`, `background-image`, `background-repeat`, `back-ground-attachment` and `background-position`. Each of these can be set individually, of course.

Background Images

As well as colored backgrounds, it's possible to set an image as the background. This can be set to repeat over the background or just be placed once:

```
body {
    background: #F4a460;;
    background-image: url("suki.jpg");
    background-repeat: no-repeat;
}
```

The HTML basically stays the same:

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>My Cat Suki</title>
```



```
    <link rel="stylesheet" type/css" href="mycss.css"
                title="green">
</head><body>
The kitchen cat, Suki!
</body></html>
```

We can see the output from this in Fig. 1. As you can see there is one image in the background with black writing on top.

If you want a repeating image in the background, possibly as a pattern for texture, then you just alter `background-repeat` to `repeat` or `repeat-x` `repeat-y` depending on which direction you want it to repeat in:

```
body {
    background: #F4a460;
    background-image: url("browntile.jpg");
    background-repeat: repeat;
}
```

The other thing that you can do is to set an image as fixed when a scroll bar is moved or moves with the rest of the page. To do this, use the `background-attachment` property, which can be set to scroll or fixed.

The summary of properties is given in Table 1.

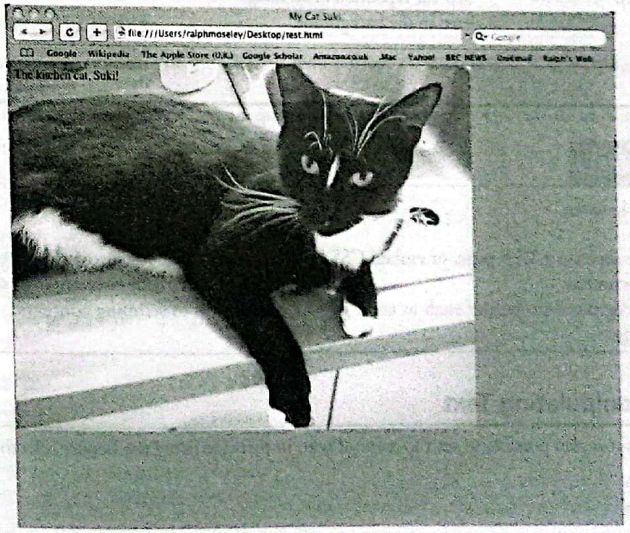


Figure 1 Using a background image.

Table 1 The background property

Property	*Description*	*Values*
`Background`	Sets all background properties in one declaration	as RGB
`Background-attachment`	Sets whether image moves with page when scrolled	scroll or fixed
`background-color`	Sets the background color of an element	RGB, hex, name or transparent
`background-image`	Sets an image in the background	URL or none

`Background-position`	Sets the starting position of an image in the background	`top left`
		`top center`
		`top right`
		`center left`
		`center center`
		`center right`
		`bottom left`
		`bottom center`
		`bottom right`
		`x-% y-%`
		`x-pos y-pos`

`background-repeat`	Sets the repetition of an image used in the background	`repeat`
		`repeat-x`
		`repeat-y`
		`no-repeat`

Checkpoint!

We now know how to:

- Link a style sheet to a Web page, or embed CSS within a document.
- Change background properties such as color or add an image.
- Manipulate a background image so it can be repeated (or tiled) over an area, or remain static in relation to scrolling contents.

5.6 Manipulating Text

As we have seen it is also possible to set the color of text, in this case using the header selector:

```
<html>
<head>

<style type="text/css">
h1 {color: #00ff00}
```



```
h2 {color: yellow}
h3 {color: rgb(255,20,40)}
p {color: rgb(0,0,255)}
</style>

</head>

<body>
<h1>A colored header 1</h1>
<h2>A colored header 2</h2>
<h3>A colored header 3</h3>
<p>And a paragraph!</p>
</body>
</html>
```

This document uses a built-in style sheet to determine the color of various parts of the text. Here, a set of headers are each given a different color, along with a separate paragraph, which is also given its own individual color. A variety of ways of choosing and setting the values are used, including giving a hex value, a name and RGB values between 0 and 255.

The background color of text can also be changed:

```
<head>
<style type="text/css">
h1 {color: #00ff00; letter-spacing: -3px}
h2 {color: yellow; letter-spacing: 0.6px}
h3 {color: rgb(255,20,40); background-color: yellow}
p {color: rgb(0,0,255)}
</style>

</head>
```

Also, in this example the spacing of the letters is manipulated on specific text. The spacing is decreased on `<h1>` and increased on `<h2>`.

When controlling layout, it is often adequate to say whether you want text to be aligned to the left, center or right. This can also be done within CSS:

```
<html>
<head>

<style type="text/css">
h1 {text-align: center}
h2 {text-align: left}
h3 {text-align: right}
</style>

</head>

<body>

<h1>This is header 1</h1>
<h2>This is header 2</h2>
<h3>This is header 3</h3>

</body>
</html>
```

CamScanner

Text Decoration

Often text needs to be underlined. This can be done with text decoration, which also has several other styles:

```
<html>
<head>

<style type="text/css">
h1 {text-decoration: overline}
h2 {text-decoration: line-through}
h3 {text-decoration: underline}
a {text-decoration: none}
</style>

</head>

<body>
<h1>This is header 1</h1>
<h2>This is header 2</h2>
<h3>This is header 3</h3>
<p>
<a href="http://www.w3schools.com/default.asp">
This is a link</a>
</p>

</body>
</html>
```

Text Indentation

Text can also be indented:

```
<html>
<head>

<style type="text/css">
p {text-indent: 1cm}
</style>

</head>

<body>

<p>
This is some text in a paragraph
This is some text in a paragraph
This is some text in a paragraph
This is some text in a paragraph
This is some text in a paragraph
This is some text in a paragraph
</p>

</body>
</html>
```


Text Case

Sometimes you may want to control the case of text in a section or paragraph:

```
<html>
<head>

<style type="text/css">
p.uppercase {text-transform: uppercase}
p.lowercase {text-transform: lowercase}
p.capitalize {text-transform: capitalize}
</style>

</head>

<body>

<p class="uppercase">
This is some text in a paragraph
</p>

<p class="lowercase">
This is some text in a paragraph
</p>

<p class="capitalize">
This is some text in a paragraph
</p>

</body>
</html>
```

5.7 Using Fonts

The following example shows many of the font features available in CSS:

```
<html>
<head>

<style type="text/css">
h3 {font-family: times;
     font-size: 150%;
     font-style: normal}
p {font-family: courier;
     font-size: 100%;
     font-style: oblique}
p.sansserif {font-family: sans-serif;
     font-style: italic}
</style>

</head>
```

CamScanner

```
<body>
<h3>This is header 3</h3>

<p>
This is a paragraph</p>

<p class="sansserif">
This is a paragraph</p>

</body>
</html>
```

The first thing to look at is how fonts are set. This is, not surprisingly, with the `font-family` property. The property `font-size` will set the size of the text, either as a percentage or a relatively worded value such as `small`, `medium`, `large`. The other property used in the example is `font-style` which allows fonts to be set to `normal`, `italic` or `oblique` (slanted).

Concept Check

- Develop a simple Web page, maybe by altering an old one with lots of HTML presentation tags in. Convert any manipulation of text via HTML, such as fonts and colors, into CSS.
- Initially embed the style detail within the same page as the HTML.
- Take out the CSS and place in a separate file.

5.8 Borders and Boxes

The border properties allow many styles of border to be placed around areas. There is a large degree of control over the type of border, such as solid, dashed, doubled or ridged. Along with this, colors, width and thickness can be applied.

The summary of properties is given in Table 2.

Table 2 The border property

Property	*Description*	*Values*
`Border`	Sets all properties for four borders in one declaration	`border-width, border-style, border-color`
`border-bottom`	Sets all properties for bottom border in one declaration	`border-bottom-width, border-style, border-color`
`border-bottom-color`	Sets bottom border color	`border-color`
`border-bottom-style`	Sets bottom border style	`border-style`
`border-bottom-width`	Sets bottom border width	`thin, medium, thick length`
`border-color`	Sets the color of the borders	`color`
`border-left`	Sets all properties for left side	`border-left-width, border-style, border-color`

(Continued)

Table 2 (Continued)

Property	*Description*	*Values*
`border-left-color`	Sets left border color	`border-color`
`border-left-style`	Sets left border style	`border-style`
`border-left-width`	Sets left border width	`thin, medium, thick length`
`border-right`	Sets all properties for right side	`border-right-width, border-style, border-color`
`border-right-color`	Sets right border color	`border-color`
`border-right-style`	Sets right border style	`border-style`
`border-right-width`	Sets right border width	`thin, medium, thick length`

`border-style`	Sets style of the four borders	`none`
		`hidden`
		`dotted`
		`dashed`
		`solid`
		`double`
		`groove`
		`ridge`
		`inset`
		`outset`

`border-top`	Sets all properties for top border	`border-top-width`
		`border-style`
		`border-color`

`border-top-color`	Sets color for top border	`border-color`
`border-top-style`	Sets style for top border	`border-style`
`border-top-width`	Sets width for top border	`thin, medium, thick length`
`border-width`	Sets width for all borders	`thin, medium, thick length`

5.9 Margins

The margin is the space between one element and another, in addition to and outside of any padding (Section 5.10) or border around an element:

```
<html>
<head>

<style type="text/css">
p.margin {margin-top: 5cm}
</style>

</head>
```

```
<body>

<p>
This is a paragraph This is a paragraph
This is a paragraph This is a paragraph
</p>

<p class="margin">
This is a paragraph with a top margin
This is a paragraph with a top margin
</p>

</body>
</html>
```

It is possible to set margins for top, bottom, left and right. You can use the margin property alone to set all the properties in one single declaration:

```
<style type="text/css">
p.margin {margin: 2cm 4cm 3cm 4cm}
</style>
```

5.10 Padding

Padding allows you to insert extra space around the contents of an element but inside the border. You can change only the padding zone's thickness but not its color.

```
<html>
<head>

<style type="text/css">
    td {padding-bottom: 2cm; padding-top: 2cm}
</style>

</head>

<body>

<table border="1">
<tr>
<td>
A table cell with top and bottom padding
</td>
</tr>
</table>
</body>
</html>
```

The summary of properties is given in Table 3.

Table 3 The padding property

Property	*Description*	*Values*
padding-bottom	Adds padding at bottom	length, %
padding-top	Adds padding at top	length, %
padding-left	Adds padding at left	length, %
padding-right	Adds padding at right	length, %
padding	All of above in one declaration	padding-top padding-right padding-bottom padding-left

5.11 Lists

There is some degree of control over the lists you create. It is possible to alter the list item marker and where the marker is placed:

```
<html>
<head>

<style type="text/css">
ul.disc
{list-style-type: disc}
ul.circle
{list-style-type: circle}
ul.square
{list-style-type: square}
ul.none
{list-style-type: none}
</style>
</head>

<body>
    <b>Pizzas!</b><br><br>
    <ul class="disc">
    <li>Vegetarian</li>
    <li>Mozzarella</li>
    <li>Al Fungi</li>
    </ul>

    <ul class="circle">
    <li>Vegetarian</li>
    <li>Mozzarella</li>
    <li>Al Fungi</li>
    </ul>
```

CamScanner

```
<ul class="square">
<li>Vegetarian</li>
<li>Mozzarella</li>
<li>Al Fungi</li>
</ul>

<ul class="none">
<li>Vegetarian</li>
<li>Mozzarella</li>
<li>Al Fungi</li>
</ul>

</body>
</html>
```

The summary of properties is given in Table 4.

Table 4 The list property

Property	*Description*	*Values*
`list-style-image`	Use an image as marker	`none, url`
`list-style-position`	Positioning of marker	`inside, outside`
`list-style-type`	Type of marker	`none, disc, circle, square, decimal ...`
`list-style`	Set all properties	`All above properties`

5.12 Positioning Using CSS

Unlike HTML, CSS allows good control over the positioning of elements. Ordinarily elements in a Web page flow in the order they appear. So, as you work down from the top of an HTML page the elements will appear: an image placed before a paragraph will appear on the page in just that way. Using CSS it is possible, instead, to break this normal flow and position elements out of this order. In fact it is possible to control exactly where elements will appear.

Absolutely!

The example in Listing 2 shows what is called absolute positioning. In this case the positioning sets the top left of the element, although it is also possible to set an element's position by the bottom right. It is labeled absolute position as it is not relative to where it would have been in the natural flow and uses the window positioning for exact coordinates.


```
<html>
<head>
<style type="text/css">
h1 {
position: absolute;
top: 100px;
left: 100px
}

p {
position: absolute;
top: 200px;
left: 100px
}
</style>

</head>

<body>
<h1>A heading</h1>
<p>
The <b>heading</b> is placed 100px
down from the top of the document,
and 100px to the right from the
left side of the document.
The <b>paragraph</b> is placed 200px
down from the top of the document,
and 100px to the right from the
left side of the document.
</p>

</body>
</html>
```

Listing 2 Absolute positioning.

It's All Relative

Elements can also be placed relatively; that is, not in relation to other objects but in relation to where they would have been in the natural flow of the page. To do this the selector `position: relative` is used, along with `top, right, bottom` or `left` and `:n` where `n` is the offset from its natural location.

The Z-Index

Not only is it possible to position elements in the x and y coordinates but it is also possible to position them in terms of depth of the screen so things can appear to be behind or in front of others! To do this,

CamScanner

a z-index is introduced. A z-index property has a number that describes where the object is in the stack of visible elements. The higher the number, the closer to the top and more visible the object is:

```
<html>

<head>
<style type="text/css">
p.ap {
position: absolute;
left: 55px;
top: 70px;
z-index: 0;
}
img.x
{
position:absolute;
left:0px;
top:0px;
z-index:1;
}
h1 {
position:absolute;
left:5px;
top:5px;
z-index: 2;
}
</style>
</head>
<body>

<h1>This is a Heading</h1>
<img class="x" src="cat.gif" width="100" height="180">
<p class="ap">Default z-index is 0. Z-index -1 has lower
                                priority.</p>

</body>
</html>
```

This example shows three layers at depths 0, 1, 2. The heading is at depth 2 so is most visible (highest in the stack), followed by the image at 1 and finally the furthest back is the paragraph. Be aware that Netscape 4 does not support the z-index.

The height and width of an element can be set quite easily too, so it can be manipulated into a given space. To do this we use the dimension properties `width` and `height`, which can be set with percentage or pixel values:

```
<style type="text/css">

p.small {
height: 30px;
```



```
width: 50px
}

p.other {
height: 50%;
}
</style>
```

In `p.small` the `width` and `height` are given of the box to contain the paragraph. In `p.other` 50% of the container area will be used for the element's box size.

If the contents of the element do not fit in the box then it is possible to use the `overflow` property to explain what you want to happen. For example, using the value `visible`, (or simply not using the `overflow` property at all) makes the box expand so that its contents will fit. The value `hidden` will hide any contents that don't fit in the box; whereas the `scroll` value will add scroll bars to the element box so that there is a choice by the user as to whether the content can be looked at. Using `auto` as the value for the `overflow` property will make scroll bars only appear when they need to.

Shaping an Element

An element, such as an image, can also be usefully shaped with the `clip` property. For example:

```
img
{
position:absolute;
clip:rect(0px 50px 200px 0px)
}
```

Floating Elements

Sometimes you want an element, such as an image, to float so that other elements will flow around it. This happens, for example, when you want a picture to be surrounded by text on one side or another. To do this the `float` selector is used together with `left` or `right` as the value. Unfortunately a lot of browsers tend to be inconsistent with whether this is implemented properly!

Layout and Structure

The idea behind CSS was to separate the formatting and styling rules from the content. It is important to maintain structure within the document that will be styled by CSS. A way to do this is to break the page into logical sections with `div` (*division*) elements. You may break a page down, for example, into banner, navigation, news and content divisions as shown in Listing 3.

```
<html><head>
    <title>A Simple Web Page</title>
    <style type="text/css">
        #banner {
            font-family: Tahoma;
            background-color: #84a563;
            padding: 2px 2px 2px 10px;
            }
```

CamScanner

```
        #navigation {
             padding: 10px;
             border: 1px dotted;
             font-family: courier;
             font-size: 100%;
             background-color: #5c892e;
        }
        #news {
             padding: 4px;
             font-family: Tahoma;
             background-color: #5c922e;
        }
        #content {
             padding: 2px;
             font-family: Tahoma;
             background-color: #84a563;
        }
        a:hover {
             font-weight: bold;
        }
        a:link, a:visited {
             text-decoration: none;
             color: #550033;
        }
    </style>
</head>
<body>
<div id="banner">
    <h1>My Special Page</h1>
</div>
<div id="navigation">
    <a href="home.html">home</a>
    <a href="resources">resources</a>
    <a href="links.html">links</a>
    <a href="feedback.html">feedback</a>
</div>
<div id="news">
    NEWS:
</div>
<div id="content">
<pre>
 Stone walls do not a prison make,
 Nor iron bars a cage;
 Minds innocent and quiet take
```



```
 That for an hermitage;
 If I have freedom in my love
 And in my soul am free,
 Angels alone, that soar above,
 Enjoy such liberty.

<i>To Althea from Prison</i>
  - Colonel Lovelace
 </pre>
</div>
</body>
</html>
```

Listing 3 Using `div` sections.

Notice in this example the various `div` sections with identifiers: these are then used to plug in or connect the desired styles.

We can see the output in Fig. 2.

Concept Check

- Write a Web page that has several blocks of text.
- Add borders, margins and padding.
- Position the text components as you want them.
- Place one on top of the other; what happens?
- Now add z-index properties and alter which one is on top.
- Try adding color or maybe a tiled image as a background.

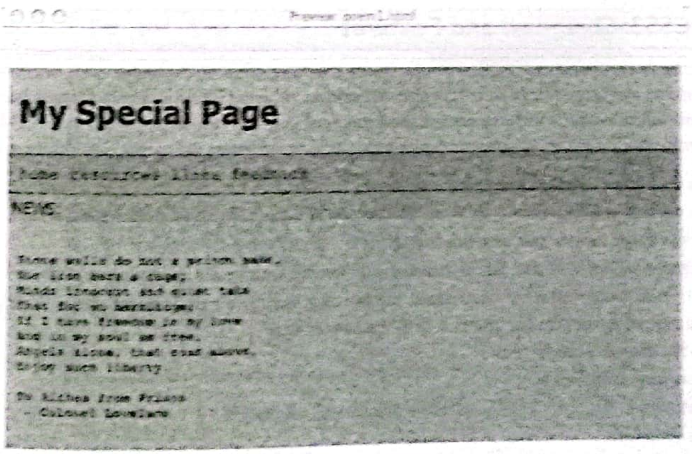


Figure 2 The finished page.

5.13 CSS2

CSS level 2 was published as a recommendation in May 1998 and has already had a revision to CSS 2.1. CSS2 mainly covered improvements to absolute, relative and fixed positioning of elements. It also concerned manipulation of media types including aural style sheets, bidirectional text and new font properties.

CSS 2.1 improves features further that are currently badly supported. It also adds already implemented browser extensions to the specification.

CSS2 aural style sheets are an interesting addition, allowing accessibility for those who are visually impaired. They allow the features of speech processing of a document to be controlled. It is possible, for example, to add features that manipulate the frequency of speech, stress, how words are to be put together and also manipulate format to be speech friendly. For example, it is possible to say where the speech process proceeds when it encounters a table; does it read data downwards or across?

CSS2 also allows print-specific instructions to be added. This allows page control during hard copy output, such as breaks, margins and doubled-sided printing instructions.

Checkpoint!

We now know:

- Display positioning of objects within a document usually occurs as a natural flow from top to bottom in an HTML document.
- It is possible to use CSS positioning to move an element relatively to its normal position or absolutely by providing pixel coordinates.
- As well as x and y positioning there is the z-index, which allows depth to be added to the display so objects can be layered and their position in this stack determined.
- CSS level 2 has further improved CSS and added features too.

5.14 CSS3: Overview and Features

The CSS3 is the latest standard of CSS and fully backward compatible with the previous versions. The CSS2 was developed as a single specification covering all its features, whereas CSS3 is divided into separate modules, where each module add new features or extend the features of CSS2. Modular approach in CSS3 leads to independent improvements and development of each module. The following are major new features of the CSS3:

- Selectors including attribute selector
- Provides various text effects
- Support for more colors and custom fonts
- Improvement in borders and backgrounds; supporting border image using `border-image` and `border-corner-image` properties; supporting multiple backgrounds for a Web page
- Supports rounded corners for the box and shadow effects
- Support for multi-column text
- Animation and transitions
- Gradients
- 2-D/3-D graphics

Let us understand some of the CSS3 features with the use of the examples.

Text Effects

The CSS3 provides various text effects with property `text-shadow`. Consider the code given in Listing 4.

```
<!DOCTYPE html>
<html>
<head>
<style>
#tstyle {
        text-shadow: 5px 10px 0px #FF0000;
}
</style>
</head>
<body>
<div id="tstyle">
                <h1>Learning CSS3</h1>
                <p>Hi, I am fine</p>
</div>
</body>
</html>
```

Listing 4 Using `text-shadow` property.

The `text-shadow` property defined in the `#tstyle` has four parameters: x-offset, y-offset, blur-radius and color having values 5px, 10px, 0px and #FF0000 (red), respectively. The `<div>` element uses `id` value `tstyle` to provide the shadow effect to all the text inside it. The output of Listing 4 is shown in Figure 3.

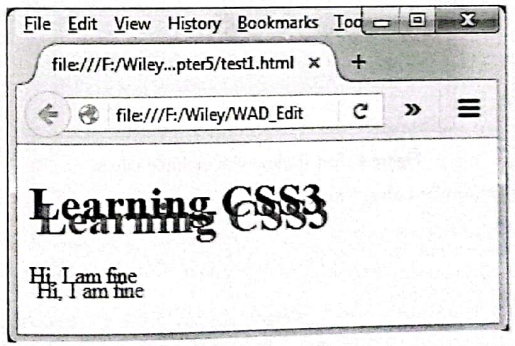


Figure 3 Text-Shadow effects.

We can also use multiple `text-shadow` effect for the text inside the `<div>` elements as shown in the Listing 5. Note that the second value in `text-shadow` has negative position values and blur-radius is 5px.

```
<!DOCTYPE html>
<html>
<head>
<style>
#tstyle {
        text-shadow: 5px 10px 0px #FF0000,
                     -5px -10px 5px #0000FF;

}
</style>
</head>
<body>
<div id="tstyle">
        <h1>Learning CSS3</h1>
</div>
</body>
</html>
```

Listing 5 Multiple `text-shadow`.

The output of the Listing 5 is shown in Figure 4.

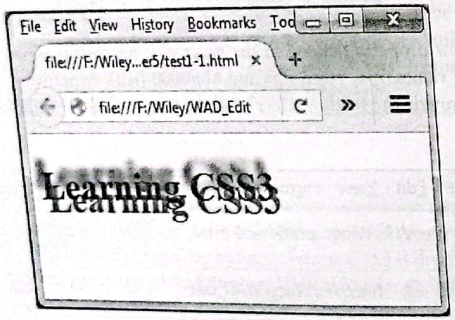


Figure 4 Text-shadow with multiple values.

If we change the style in the Listing 5 to

```
#tstyle {
        text-shadow: 0px 0px 5px #FF0000;
        background: #000000
        color: #00FF00;
}
```

then the text "Learning CSS3" is displayed with green color in black background and red blurring as header1.

Rounded Borders

The CSS3 provides many useful properties for the borders, specifically to create the rounded borders. Listing 6 produces the output as shown in Figure 5.


```
<!DOCTYPE html>
<html>
<head>
<style>
#rnd {
    border-radius:25px;
    border: 2px solid blue;
    padding: 20px;
    width: 200px;
    height: 100px;
}
</style>
</head>
<body>
    <p id="rnd">Box With Rounded corners</p>
</body>
</html>
```

Listing 6 Using `border-radius`.

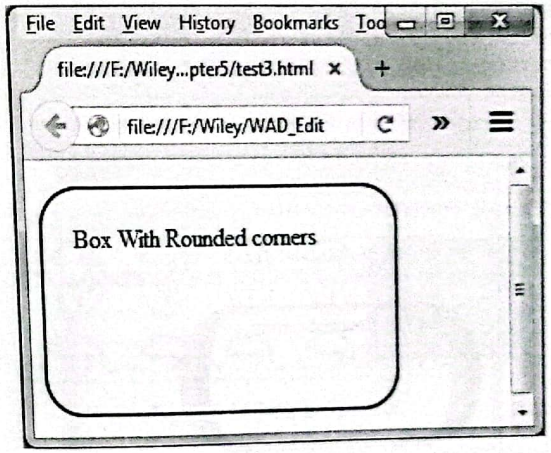


Figure 5 Rounded corners.

The preceding example (Listing 6) uses same value for all four corners of the box that is 25px with `border-radius` property. It is possible to use four different values for `border-radius` property in order of top-left, top-right, bottom-right and bottom-left. For example, if we use border-radius in Listing 6 as

```
border-radius: 10px 50px 10px 50px;
```

then the box will appear as shown in Figure 6.

CamScanner

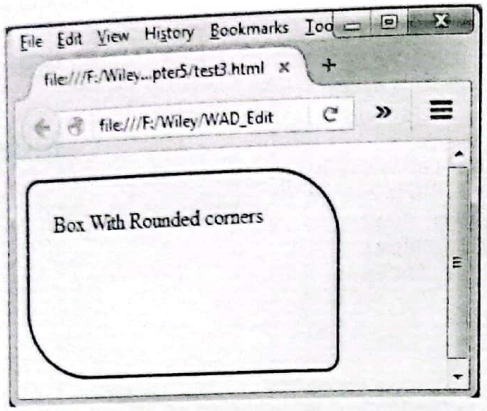


Figure 6 Rounded corners with different radius values.

Figure 7 shows the rounded box with the use of following style.

```
#rnd {
        border-width: 5px 25px 5px 25px;
        border-radius: 50px;
        border-style: groove;
        border-color: #FF0000;
        width: 200px;
        height: 100px;
}
```

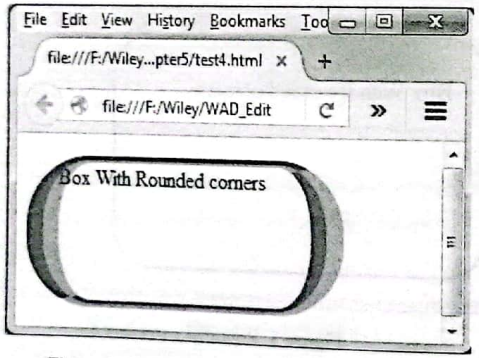


Figure 7 Box with style and rounded corners.

Multiple Backgrounds

The background properties are very useful for the Web designers to provide backgrounds. With the support of multiple backgrounds, CSS3 provides very simple and easy way with better control to Web designer. Listing 7 uses two different images as backgrounds with one repeated (face.jpg) and one placed only on right (plane.jpg).


```
<!DOCTYPE html>
<html>
<head>
<style>
#multi_back {
        background-image: url(plane.jpg), url(face.jpg);
        background-position: right, left top;
        background-repeat: no-repeat, repeat;
        padding: 20px;
}
</style>
</head>
<body>
   <div id= "multi_back">
        <h1>Learning CSS3</h1>
        <p>It is fun to learn CSS3</p>
        <p>HTML5 and CSS3 together provides Web designer all
           they wants</p>
   </div>
</body>
</html>
```

Listing 7 Using multiple backgrounds.

When displayed in browser, the output of Listing 7 is seen as shown in Figure 8.

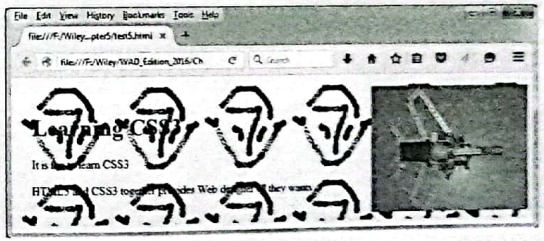


Figure 8 Multiple backgrounds.

Gradients

The CSS3 gradients are very useful feature allowing smooth transitions between two colors. The CSS3 supports linear and radial gradients. The linear gradients provide the transition in specified direction, whereas radial gradient provides transition from the center. The syntax for the linear gradient is as follows:

```
liner-gradient(direction, color1, color2, ...);
```

The default direction is top to bottom. For Mozilla Firefox browser, the `-moz-linear-gradient` is needed to be used. Listing 8 produces the background providing gradient with transition from yellow to red color.

CamScanner

```
<!DOCTYPE html>
<html>
<head>
<style>
#mygrad {
    background: yellow; /* If gradient not supported */
          background: linear-gradient(yellow, red);
    background: -moz-linear-gradient(yellow, red);
}
</style>
</head>
<body>
    <div id="mygrad">
          <h1>Learning CSS3</h1>
          <p>It is fun to learn CSS3</p>
          <p>HTML5 and CSS3 together provides Web designer
             all they wants</p>
    </div>
</body>
</html>
```

Listing 8 Using `linear-gradient`.

Figure 9 shows the effect of the Listing 8 when displayed by browser.

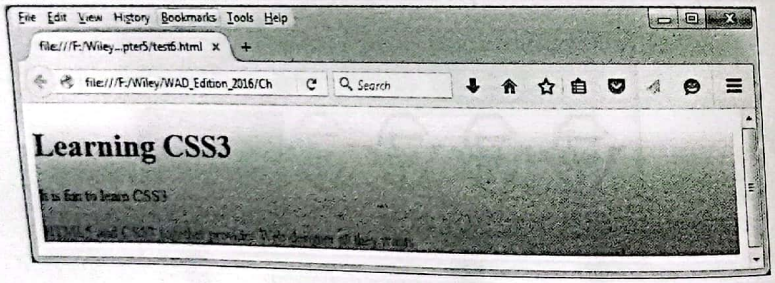


Figure 9 Linear gradient.

Following style is used to create the gradient from left to right with color stops yellow and red.

```
#mygrad {
    background: yellow; /* If gradient not supported */
    background: linear-gradient(left,yellow,red);
    background:-moz-linear-gradient(left,yellow,red);
}
```

To create a gradient in diagonal direction, following style (starts from left-top) can be used.


```
#mygrad {
    background: yellow; /* If gradient not supported */
    background: linear-gradient(left top,yellow,red);
    background:-moz-linear-gradient(left top,yellow,red);
}
```

Following is an example of gradient with multiple color stops.

```
#mygrad {
    background: yellow; /* If gradient not supported */
    background: linear-gradient(yellow, red, blue);
}
```

The CSS3 also support repeating gradients with property `repeating-linear-gradient`. Following is an example, which when displayed produces the output as shown in Figure 10.

```
#mygrad {
    height: 200px;
    background: yellow; /* If gradient not support */
    background: repeating-linear-gradient(yellow 10%, red 20%);
}
```

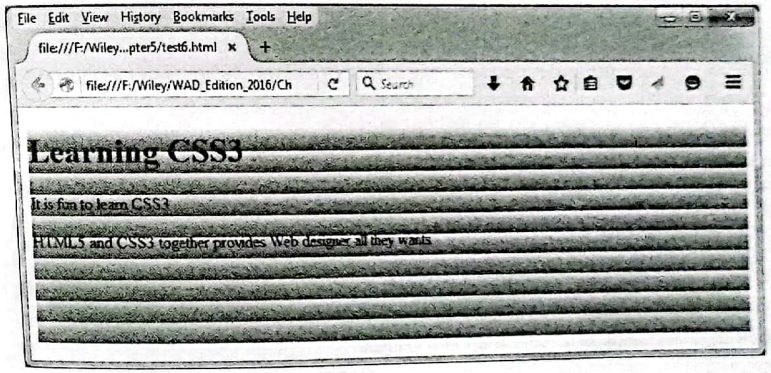


Figure 10 Repeating linear gradients.

The direction values are predefined and limited. If you want finer control on the direction, use the angle instead of direction, for example, -90deg. CSS3 also supports radial gradient which starts from the center. Let us see its effect with example. The following style creates the redial gradient with evenly spaced color stops as shown in Figure 11.

```
#mygrad {
    width: 300px;
    height: 200px;
    background: radial-gradient(yellow,blue,red);
}
```

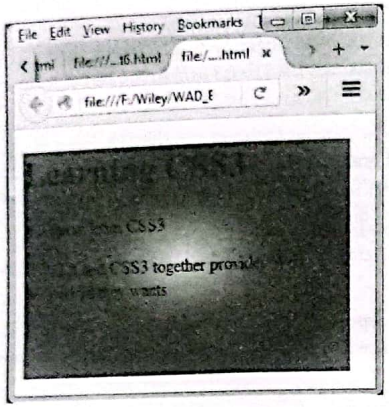


Figure 11 Radial gradient.

To create the redial gradient with the differently placed color stop, we can use the style as given below.

```
#mygrad {
    width: 300px;
    height: 200px;
    background: radial-gradient(yellow 10%,blue 20%,red 30%);
}
```

It is also possible to use the shape in radial gradient having default value ellipse. The following radial gradient uses the circle shape.

```
#mygrad {
    width: 300px;
    height: 200px;
    background: radial-gradient(circle,yellow,blue,red);
}
```

Following is an example of the repeating radial gradient.

```
#mygrad {
    width: 300px;
    height: 200px;
    background: repeating-radial-gradient(circle,yellow,blue 5%,red
      10%);
}
```

There are also other useful features associated with the gradients like transparency which makes it very powerful for creating modern Web sites.

Apart from the above features which are discussed with the examples, CSS3 supports very useful and rich set of features like borders with images, custom fonts, multiple columns, animations, 2D/3D graphics etc. The HTML5 and CSS3 together enable the Web designers to design modern Web sites in simple and easy manner.

Checkpoint!

We now know that:

- CSS3 is the latest version of CSS with many advanced features.
- CSS3 consists of modules which include new modules and extension of CSS2 features.
- CSS3 provides new features like more colors, custom fonts, text effects, borders and backgrounds, multiple columns, gradients, animations, 2D/3D graphics and many more.
- HTML5 and CSS3 together make Web very powerful for developing modern Web sites like Web 2.0.

Summary

1. Initially HTML had a large number of presentation and formatting features.
2. The need for CSS was based on the separation of presentation from HTML.
3. HTML was redefined to be concerned with structural markup.
4. CSS works by providing a selector and a set of rules to be applied.
5. The selector identifies a particular element or logically defined section.
6. A pseudo-class allows links to have styles applied.
7. A style sheet can be embedded within the page or in a separate file with the .css extension, then linked in.
8. A style sheet can control background images, colors and properties.
9. Text can also be manipulated in terms of font, size, decoration, alignment and spacing.
10. Style sheets can also control borders and various spacing possibilities around an element or section.
11. Lists are capable of being styled.
12. CSS can control positioning more precisely, both absolutely and relatively.
13. Using the z-index feature it is possible to control positioning in terms of the order of items when overlapped. This adds a depth control to the display.
14. Images can be clipped and manipulated.
15. Elements can be floated so text flows around them.
16. The `div` element provides the ability to break HTML into logical elements that can be manipulated using CSS.
17. CSS2 has improved CSS further and enhanced many features while providing some that are new, such as aural style sheets.
18. CSS3 is the latest version of the CSS and is developed as modules.
19. CSS3 features like text effects, rounded borders, multiple backgrounds and gradients are very useful for creating modern Web sites.
20. CSS3 also supports features like custom fonts, animations, 2D/3D graphics, multiple columns etc.

Key Words and Phrases

Absolute When applied to CSS positioning, refers to the specifying of exact coordinates for an element.

Border Defines all aspects of a border.

Class For giving a name to each class of divisions.

CSS3 A major revision of the CSS that provides the features to make the Web sites more visually appealing.

CamScanner

Div Divides a page into logical sections.

ID For giving a unique name to a particular division.

Margin For setting the amount of space between the sides of an element's border and the parent element.

Padding For specifying the distance between the sides of an element's content area and the border.

Relative When applied to CSS positioning, refers to specifying coordinates that are relative to the natural positioning of an element within a Web page.

RGB Red Green Blue, a model used for color definition.

z-index Allows the setting of the depth of an element in relation to the overlapping of other elements.

Multiple-Choice Questions

1. The CSS was introduced to handle the _____ part within HTML.
   (a) Presentation
   (b) Structure
   (c) Both of the above
   (d) None of the above
2. Which of the following types of CSS can be used with multiple Web pages?
   (a) Internal
   (b) Inline
   (c) External
   (d) All of the above
3. Which of the following tags is used to connect the external CSS to your Web page?
   (a) `<connect>`
   (b) `<link>`
   (c) `<style>`
   (d) None of the above
4. Which of the following is an example of inline style sheet?
   (a) `<style> h1 {color:red} </style>`
   (b) `<h1 style="color:red"> ... </h1>`
   (c) `<body bgcolor="#800000">`
   (d) None of the above
5. Which of the following denotes id in case of CSS?
   (a) `p.right { ... }`
   (b) `.left { text-align:left }`
   (c) `p#bluepara { ... }`
   (d) None of the above
6. Which of the following attributes is used to make text underlined in an HTML element?
   (a) `text-underline`
   (b) `text-decoration`
   (c) `text`
   (d) None of the above
7. Padding means the extra space
   (a) Between content and border
   (b) Between two elements
   (c) Depends on how it is defined
   (d) None of the above
8. Which of the CSS properties is used to control the size of the text in Web document?
   (a) `text-size`
   (b) `font-size`
   (c) `text-style`
   (d) None of the above
9. Which of the following CSS positioning is used to place an element at particular point in screen?
   (a) Absolute
   (b) Relative
   (c) z-index
   (d) All of the above
10. Which of the following CSS position methods decides the depth of an element?
    (a) Absolute
    (b) Relative
    (c) z-index
    (d) All of the above

Review Questions

1. What is CSS? What is the need of the CSS?
2. Explain the basic syntax of the CSS with example.
3. What is class selector in CSS? Show its use with example.
4. What is id selector in CSS? Show its use with example.
5. Compare class and id selectors in CSS with proper illustration.
6. What are different types of Cascading Style Sheets? Elaborate them with examples.
7. What is external style sheet? Give an example. List its advantages.
8. Differentiate between internal and inline style sheets giving example.
9. Explain the use of the pseudo-classes with anchor tag. Give a proper example.
10. Write an external style sheet for the following rules:
    (a) To make all the headers with normal font and having size 150%.
    (b) To display all the paragraphs in Arial font with bold style.
    (c) To make the background color red.

    Show the use of above CSS with HTML file by linking it.
11. Explain and show the use of background property to make your background attractive using CSS.
12. Write a style sheet for making background having color red with image (bk.jpg) which is repeated.
13. Explain the use of the text and font properties to make your text having different presentations using CSS.
14. Consider a following CSS.

```
<style type="text/css">
p.margin {
margin: 2cm 2cm 1cm 2cm;
border: thick solid red;
padding-top: 0.5%
}
</style>
```

    Give its meaning and effect when it applies to a paragraph in your page.
15. Show the use of absolute and relative positioning with CSS by giving an example.
16. What is z-index in CSS? Show its use with example.
17. What is a `div` element?
18. What is CSS3? List its features and explain any one.
19. Discuss the text effects with example.
20. How can you provide rounded border to a box? Give an example.
21. What is gradient? Explain linear and radial gradients with examples.

Practice

1. Write an internal CSS with following rules.
   (a) Paragraph with font-family times, size 150%.
   (b) Text capitalized and underlined with indent of 1 cm.

   Make its use in body part of your sample html page.
2. Write an external CSS with following rules.
   (a) List of items having marker circle.
   (b) Each item of the list having top and right solid border.
   (c) Margin between two items is 1 cm.

CamScanner

(d) Padding with its top border for each item being 2%.

Make the use of above CSS in your Web page by linking it.

3. Develop a Web page to display your one page resume. For all the HTML tags used in your page, write CSS for appropriate presentation and use it.
4. Develop a login Web page with application of external CSS so that on all the pages of your Web site, the login box is having consistent style.
5. Develop a static Web site having 5 to 10 pages to give information regarding your college such that
   (a) It uses all the important HTML tags.
   (b) It uses all types of CSS for presentation and style.
   (c) It is consistent in design with having your own Web site theme.
   (d) It uses the effective navigation style.
   (e) There are proper page layouts.
6. Visit Web resources like www.w3scools.com and study and practice the advanced features of the CSS.
7. Download the ready-to-use CSS files from the Web and see their effects by applying them in your test Web pages.
8. Develop a Web page for a magazine having an article with multiple columns using CSS3.
9. Use CSS3 features to create an animation in your Web site.
10. Apply CSS3 features to develop the Web site for Practice 5.

Chapter Quiz

1. One of the best ways to learn CSS is to actually experiment with using it. Take a previous design or Web page that was developed in HTML and redevelop it using CSS.
2. The Zen Garden project is extremely interesting (see `http://www.csszengarden.com/`). The idea is that you download their HTML page and develop your own CSS to give it a new presentation. First of all look at other people's attempts at showing off CSS, then try and see what you can come up with.

Useful Web Addresses

http://www.csszengarden.com/
http://en.wikipedia.org/wiki/Cascading_Style_Sheets/
http://www.w3schools.com/css/
http://www.w3.org/TR/REC-CSS1-961217.html
http://www.w3.org/TR/REC-CSS2/

Answers

1. (a)
2. (c)
3. (b)
4. (b)
5. (c)
6. (b)
7. (a)
8. (b)
9. (a)
10. (c)

6 JavaScript: Introduction to Client Side Scripting

LEARNING OBJECTIVES

At the end of this chapter, you will be able to:

- Define client side scripting and understand its advantages and disadvantages.
- Embed JavaScript code into an HTML document using script tag.
- Use the external JavaScript code into HTML pages.
- Define and use JavaScript variables for various data types.
- Define and use assignment statement.
- Work with strings and arrays in JavaScript.
- Write and use user-defined functions.
- Define and use conditions, switch statement and looping statements.

This chapter will prepare you for developing with this popular scripting language, showing you the syntax and possibilities of use. The aims here are to show you how it is placed within a page, variables, strings, arrays and loops. Program flow is also discussed and how conditional operators and commands are used.

This chapter gives you an initial grounding in JavaScript, exploring why it is needed, where it came from and its basic language structure. It also looks at several important differences between JavaScript and other programing languages and how to start developing dynamic Web pages for interactivity with users.

Scripting may sound like a diminished aspect of programing but JavaScript is a complex, fully featured programing language. Like any programing language it may appear to be fairly simple but in fact it can be more complex than most. If there is not a good basic understanding it can be very frustrating to write even simple programs.

There is a myth too that JavaScript is in some way related to Java. This is in fact incorrect and, although there are some similarities, there is a great deal of difference between the two, as will be shown in this chapter. JavaScript does use objects but there is some argument as to whether it can in fact be called object oriented rather than object based.

JavaScript has been around for many years and has quite a history, but this chapter will explore how it has matured to become a sophisticated companion to any Web designer who wants to make their pages rich and dynamic.

6.1 What is JavaScript?

HTML and CSS concentrate on a static rendering of a page; things do not change on the page over time, or because of events. To do these things, we use scripting languages, which allow content to change dynamically. Not only this, but it is possible to interact with the user beyond what is possible with HTML.

CamScanner

Scripts are programs just like any other programing language; they can execute on the client side (the one with the browser) or the server. In these initial chapters we will concentrate on the client side before moving on to the server.

It is important to realize there are advantages to running scripts on either side. For example, running a script on the client saves processing time on the server (and vice versa!). Doing some processing on the client before sending information may be advantageous: you could, for example, check information before it is sent.

These are the advantages of client side scripting:

1. The Web browser uses its own resources, and eases the burden on the server.
2. It has fewer features than server side language.

These are the disadvantages of client side scripting:

1. Code is usually visible.
2. Code is probably modifiable.
3. Local files and databases can't be accessed (as they are stored on the server).

The main two client side scripting languages are JavaScript and VBScript. The more popular of the two is probably JavaScript. There is also JScript, a Microsoft version of the same language. The need for a common specification led to ECMA 262 and the development of a standardized version of the language called ECMAScript.

JavaScript was originally called LiveScript (and before that Mocha!), and was developed by Netscape Communications. Despite the name it has little in common with the programing language Java, which sometimes leads to confusion. JavaScript is an object-based language, rather than an object-oriented one, based on the idea of prototypes. The main similarity is that they share common syntax, which bears some relation to C.

JavaScript is embedded in Web pages and interpreted by the browser.

6.2 How to Develop JavaScript

It's possible to develop JavaScript programs using a simple editor and your browser in the same way as you would HTML and CSS. It's also possible to use IDEs such as Dreamweaver, which can help with development and speed up your writing of client side scripts.

There are a large number of JavaScript programs on the Web, so it's not always necessary to write your own and reinvent the wheel. You may like to modify someone else's code for your requirements or study how they did it.

JavaScript can be used to make your site more interactive and liven up your pages, creating an active user interface. You can validate incoming information from forms and improve the way your navigation interface works. JavaScript has the ability to create HTML pages on the fly, which could be dependent on choices that the user makes.

6.3 Simple JavaScript

There are a few options as to where you place your JavaScript within the HTML; it can also be external to the HTML but linked in.

Embedded

JavaScript can be embedded in an HTML document. The media type for JavaScript is `application/x-javascript` but, more commonly, to embed it in HTML you must begin it with:

```
<script type="text/javascript">
```

and end with:

```
</script>
```

You may also see code for older browsers begin with:

```
<script language="Javascript" type="text/javascript">
<!-
```

and end with:

```
//-->
</script>
```

The script tag has the effect of stopping the JavaScript being printed out as well as identifying the code enclosed. The JavaScript can be placed in the head section of your HTML or the body.

```
<html>
<head>
      <title>A small piece of JavaScript</title>
</head>
<body>
      <script type="text/javascript">
            document.write("hello, JavaScript user!");
      </script>
</body>
</html>
```

Listing 1 A simple JavaScript program.

Listing 1 shows a complete, but small, example program. As you can see it is embedded in an HTML page within the body section. The script is first of all identified with the `<script>` tag. The only piece of JavaScript here is the output to the browser, using the `document.write` statement.

Scripts placed in the body section are executed as the page loads and can be used to generate the content of the page.

As well as the body section, JavaScript can also be placed in the head part. The advantage of putting a script in there is that it loads before the main body. This makes sure it is loaded before anyone wishes to use it, so it would be good for functions, for example, or event driven programing.

External Scripts

If you want to use the same script on several pages it could be a good idea to place the code in a separate file, rather than writing it on each. That way, if you want to update the code, or change it, you only need to do it once.

CamScanner

Simply take the code you want in a separate file out of your program and save it with the extension `.js`. The script tag is not needed in the external file, just the code. The JavaScript must then be linked to at the point you want it to be inserted in your HTML. If the script was saved as `myscript.js`:

```
<html>
<head>
</head>
<body>
<script src="myscript.js"></script>
</body>
</html>
```

Checkpoint!

- JavaScript has a history that stretches back to the early days of the Web.
- There are advantages to running scripts on the client side.
- JavaScript is the main client side scripting language.
- JavaScript has been standardized as ECMAScript.
- JavaScript can be developed with simple software tools such as an editor and browser, or using an IDE.
- It can be embedded in a Web page inside the body or head section.
- It can also be linked in as an external script.

6.4 Variables

As in many programing languages, JavaScript lets you store information in named variables, which are just storage memories so you can access the data later by using a name you have assigned. A variable's value can change as the script runs too.

A variable can contain several types of value:

1. number – a numeric value, e.g. 156, 1000, 1.2
2. string – characters wrapped in quotes, e.g. "Cat", "The house is red"
3. boolean – a value of true or false
4. null – an empty variable
5. function – a function
6. object – an object

These are known as primitive data types, with only a single value. JavaScript also supports a composite data type known as an object. An object represents a collection of values, ordered or unordered. In JavaScript an object can also be a function; that is, an object with some associated code that is attached to it.

A JavaScript variable name has a few other attributes to take note of

1. It is case sensitive – `mynum` and `MyNum` are different variables.
2. It cannot contain punctuation, spaces or start with a digit.
3. It cannot be a JavaScript reserved word.

An important difference between languages such as C and Java is that variables can contain any type and also change between types through the course of the program. Although you can declare a variable with the `var` statement you do not have to give a type anywhere to make it only hold a specific kind of value. This property makes the language *untyped*.

Scope

A JavaScript variable can have either local or global scope; however, this is not quite the same as other languages that have block-level scope. Block-level variables are, as the name suggests, contained and not accessible outside of the block they were created in.

If a variable is created with the `var` statement such as

```
var a;
```

or

```
var a,b,c;
```

then the scope will either be global (available throughout the program) or local depending on where declared. JavaScript contains, like C, functions that can be called from anywhere in the main program. If a variable is inside a function then it will be local to that function and only available there. If declared in the main program then it will be available throughout the program.

If `var` is not used at all then the variable will have a global scope whether in a local function or not. It's possible to have a global variable and a function (local) variable with the same name and they will not interfere with each other, although inside the function only the local version will be available.

```
<html>
<body>

<script type="text/javascript">
var myname = "Fred";

tester();
document.write(myname);
tester();
document.write(othername);
document.write(hername);

function tester(){
var myname="Bert";
var hername="Jane";
othername="Sid";
document.write(myname);

}
</script>
</body>
</html>
```

Listing 2 Script showing use of scope for variables.

Try and follow the program in Listing 2 by dry running it yourself without the computer. The output from it running is

```
BertFredBertSid
```

This should demonstrate the various scopes quite well! Notice in particular that Jane never gets printed as it does not exist outside the function.

CamScanner

Concept Check

- Write a simple JavaScript program as your first attempt at scripting; add code to the head and body sections of a Web page.
- Now take out the JavaScript code and put it into an external script and link it into an HTML file.
- Write a script to make sure you understand how the scope works in JavaScript, using the program in Listing 2 as a starting point.

Assignments

As we can see from Listing 2, variables are assigned their values with the equals operator '='. This is true for all types of value, whether it be string or numerical data. So, the X could be set to 123 or "Bert" by

```
X = 123;
X = "Bert";
```

If X was made equal to 123 how would you add one?

```
X = X + 1;
```

would work but so would

```
X++;
```

which means increment the value. You can also use

```
X--;
```

to subtract. There are other short cuts too, for example `x = x + y` can also be stated as `x += y`. Table 1 shows the various assignments possible.

Strings

A string can be defined as a sequence of letters, digits, punctuation and so on. A string in JavaScript is wrapped in single or double quotes. There is no special type for a single character like in C (with `char`); it is simply a string with a length of one. When you assign a string it must only take up one line and not be broken up over several and if you want to include a newline character you would use `\n` which is similar to C, C++ and Java.

Table 1 Special assignment operators

Assignment	*Equivalent to*
X+=Y	X=X+Y
X-=Y	X=X-Y
X*=Y	X=X*Y
X/=Y	X=X/Y
X%=Y	X=X%Y

Table 2 Escape sequences

Sequence	*Character*
\t	Tab
\n	Newline
\r	Carriage return
\"	Double quote
\'	Apostrophe or single quote
\\	Backslash

You can incorporate HTML into your strings, possibly for output to the browser. It must be remembered that some characters that you would like to include in a string need special attention and just like newline need a backslash to 'escape' them. For example, if you want to include a single or double quote within the string you would have to use a backslash before it – in this case to say to JavaScript that this is actually not the end of the string!

A few more useful escape sequences are shown in Table 2.

Strings can be joined together with the + operator, which is called *concatenation*. For example, it is possible to

```
mystring = "my name is" + "fred";
myother = "my name is" + name;
```

where `name` is a string variable with possibly a name as its contents.

As a string is an object type it also has some useful features. For example, if you wanted to find the length of a string you can use the `length` property:

```
lenStr = mystring.length;
```

There are also a number of methods available. If you wanted to get a specific character from a string you could use this:

```
nChar = mystring.charAt(3);
```

This would return the third character. There are other methods that will extract a substring or find the position of a letter in a string, as shown in Table 3.

Arrays

An array is a collection of data. Each item in the array has an index to access it with and they begin at zero. So, the first item will be at `i[0]` if `i` is the name of the array, whereas the fourth item will be at `i[3]`.

Arrays are created using the `Array( )` constructor:

```
var myArray = new Array( );
a[0] = 1;
```

CamScanner

Table 3 Some methods attached to the string object

Method	*Description*
`charAt`	Return the character at a specific index
`indexOf`	Find the first index of a character
`lastIndexOf`	Find the last index of a character
`substring`	Return a section of a string
`valueOf`	Return the numeric value of a string
`toLowerCase`	Convert a string to lower case
`toUpperCase`	Convert a string to upper case

```
a[1] = 2;
a[2] = "three";
a[3] = false;
```

Notice here that any type can be assigned.

It's also possible to define initial values with the constructor:

```
var myArray = new Array(1, 2, "three", false);
```

Or, you can declare the number of elements that the array will have

```
var myArray = new Array(15);
```

Concept Check

- Use some of the inbuilt functions of strings to search for a word within a sentence. It should then say whether the word is present.
- How would we convert an input string to all lowercase before validation?

Checkpoint!

- JavaScript uses variables similar to other languages but they are untyped.
- Although the type is undeclared there are specific rules for their names.
- There are scope rules for such variables, which are determined by position within the script and the use of the `var` command.
- Variables can be assigned to with the equals operator and some other special statements such as +=, *=. Variables can be incremented and decremented using ++ and --.
- Strings are classed as an object and can be manipulated with the built-in methods and properties.
- An array is an object that also has built-in properties and methods for manipulating its data.

6.5 Functions

A function is a section of code that is separate from the main program. It is defined once but can be invoked many times. A function can be passed as parameters so that they can be used and a value can be returned back. There are some functions already built in to JavaScript, such as the `Math.cos( )` function, which calculates the cosine of an angle.

An example function could be:

```
function multByTen(x)
{
   return x*10;
}
```

This can then be invoked by using the function's name complete with any parameters you want to pass:

```
mysum = multByTen(3);
```

```
<html>
<body>
 <script type="text/javascript">
 var z = multXbyY(10, 15);
 document.write("The result is "+z);
 function multXbyY(x,y) {
   document.write("x is "+x);
   document.write("y is "+y);
   return x*y;
 }
 </script>
 </body>
 </html>
```

Listing 3 Script showing use of function with parameters.

Listing 3 shows how the function can be placed at the end of the main code. It will only execute when called though. In this script two parameters are passed.

One feature of JavaScript that is different to some other languages is that the function is a data type in itself. So, it is possible to pass a function to other functions or store them in variables. They can also be assigned to object properties like other values can and become a method of that object.

6.6 Conditions

When you want to do something depending on the value of a variable then the `if` statement is one way of approaching the solution. The syntax is much the same as a lot of programing languages:

```
if (a==10) {
// execute this block if true
}
```

CamScanner

Table 4 Conditional operators

Operator	*Description*
<, <=	Less than, less than or equal to
>, >=	Greater than, greater than or equal to
!	Logical complement
!=	Test for inequality
==	Test for equality
===	Test for identity
!==	Test for non-identity
&&	Logical and
||	Logical or

The condition usually requires an operator which in this case is the check for equality; in JavaScript there are quite a few with many uses, as shown in Table 4.

The useful addition to an `if` statement is an `else`:

```
if (a==10) {
// execute this block if true
} else {
// do this if not true
}
```

Switch

The `switch` statement is another useful construct that works on a conditional basis. It works well when there are a small number of set choices:

```
switch (expression) {
    case label1:
        // code to be executed if expression == label1
    break;
    case label2:
        // code to be executed if expression == label2
    break;
    case label3:
        // code to be executed if expression == label3
    break;
    default:
        // code to executed if all the above do not match
}
```

An example of use is shown in Listing 4.


```
<html>
<head></head>
<body>
<script type="text/javascript">
//This finds the current Month 0-11
//depending on this decides on a
//message
var m=new Date( );
theMonth=m.getMonth( );
document.write(theMonth);
switch (theMonth)
{
case 4:
  document.write("The Merry Month of May");
  break;
case 0:
  document.write("Cold January!");
  break;
case 6:
  document.write("Summer Time!");
  break;
default:
  document.write("When is it holiday time?");
}
</script>
</body>
</html>
```

Listing 4 Script showing use of `case` statement.

Conditional Operator

As in other languages JavaScript also contains a conditional operator:

```
avariable = (condition) ?value1:value2;
```

An example of use would be:

```
hello=(lang=="french")?"bonjour":"hi";
```

In this case the variable `hello` is either set to `bonjour` or `hi` depending on how the variable `lang` is set. If it is set to `french` it will choose the french greeting, `bonjour`; if not it will choose `hi`.

6.7 Loops and Repetition

It's often the case when programing that you need to iterate (repeat) over a section of code several times. You may want to repeat a set number of times, or break out of the loop when something happens, or go on forever (unlikely, but possible, if you really want to!).

CamScanner

Table 5 JavaScript loop structures

Loop Construct	*Description*
`for (initialize; condition; increment) {}` `for (x=0; x<10; x++) {}`	Use when you know how many repetitions you want to do
`while (condition) {}` `while (x<10) {}`	Loop through block of code while condition is true
`do {} while (condition)` `do {} while (x<10)`	Execute block at least once then repeat while condition is true

The `for` loop works by initializing a variable to a start value, checking for a condition which, while true, the loop will run and finally incrementing (or decrementing) a value. If you had an array you wanted to print out between 10 and 20 then you could:

```
for (i=10; i<=20; i++) {
    document.write(x[i]);
}
```

Using a `while` loop to do the same thing you could:

```
i=10;
while (i<=20) {
    document.write(x[i]);
    i++;
}
```

Finally, the `do while`:

```
i=10;
do {
    document.write(x[i]);
    i++;
} while (i<=20);
```

Table 5 shows the different kinds of loop construct available in JavaScript.

Concept Check

- Write a script that uses functions to calculate how many days it is to your birthday.
- Extend the above programs to output various messages depending on how long there is to go and a special message if it is your birthday.
- Alter the program so it will repeat through several birth dates for different people and tell you how long to go for each.
- Using a loop, search for a word in sentence held in an array, returning the index. This is similar to some inbuilt JavaScript functions.

Checkpoint!

- JavaScript allows the use of functions.
- Functions, although defined once, can be invoked many times.
- It can be passed parameters and also return values.
- The function is a data type in JavaScript, unlike other languages, and can be passed as an argument to other functions.
- It can also be assigned to object properties.
- There are several conditional language structures such as `if {} else`, `case` and the conditional operator.
- There are also several loop structures such as the `for`, `while` and `do {} while`.

6.8 JavaScript Popup Boxes

One of the purposes of the JavaScript is to make the Web pages interactive. This is possible using the three different popup boxes JavaScript provides. They are listed as follows:

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

The alert box is used to inform or give a message to the user. The syntax of the alert box is

```
alert("Web page is loaded!");
```

When the above alert box pops up, it looks like as shown in Fig. 1. Press the "OK" button to proceed further. The confirm box is used to get the user confirmation for certain action. The syntax of the confirm box is

```
var c = confirm("Are you sure?");
```

When the above confirm box pops up, it looks like as shown in Fig. 2. If you press "OK" button, the value of variable c is true and if you press "Cancel" button, the value of variable c is false. The prompt box is used to receive a value from the user. The syntax of the prompt box is

```
var val = prompt("Enter an Integer");
```

When the above prompt box pops up, it looks like as shown in Fig. 3.

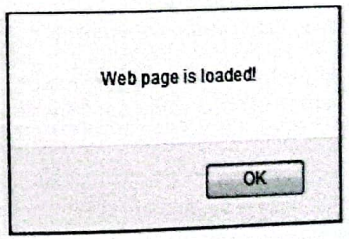


Figure 1 Alert box.

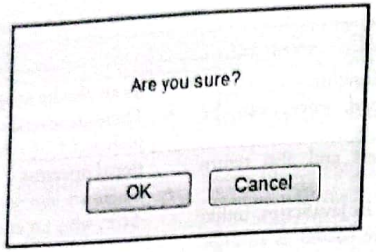


Figure 2 Confirm box.

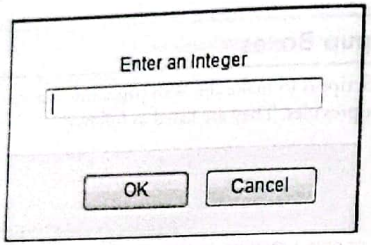


Figure 3 Prompt box.

Enter the value and press the "OK" button to store the given value in variable `val`. Pressing "Cancel" button returns null. We can also provide the default value as second argument in the following manner:

```
var val = prompt("Enter an Integer",0);
```

This will be displayed automatically when prompt box pops up. Listing 5 shows the use of all of the above popup boxes. Run it in your browser to see the working.

```
<html>
<body>
<script type="text/javascript">
alert("Welcome to JavaScript");
var c = confirm("Are you ready to work?");
if(c == true) {
        var name = prompt("Enter your name","Mansukh");
        document.write("Your name is "+name);
}
</script>
</body>
</html>
```

Listing 5 Script showing the use of popup boxes.

6.9 JavaScript and HTML

We know that the HTML alone can provide only static Web pages which do not change dynamically once loaded. The JavaScript overcomes this limitation and adds dynamism to the Web pages by allowing them to change dynamically by running the script code at client side only. The following listings demonstrate how we can use JavaScript with the HTML to make the Web page dynamic.

Listing 6 shows how we can use JavaScript code to generate the HTML tags. It simply generates the list of the books with a header. Figure 4 shows the output of Listing 6.

```
<html>
<body>
<script type="text/javascript">
        document.write("<h1>List of the Books</h1>");
        document.write("<ul>");
        document.write("<li>Developing Web Applications</li>");
        document.write("<li>Advance Java Technology</li>");
        document.write("<li>8086 Programming</li>");
        document.write("</ul>");
</script>
</body>
</html>
```

Listing 6 Display list of books.

Listing 7 performs the same task as Listing 6. However, Listing 7 is more dynamic in the sense that it reads the data from the array and displays it in form of the list. To add more data, you simply need to add them into the array. Listing 7 produces the same output as shown in Fig. 4.

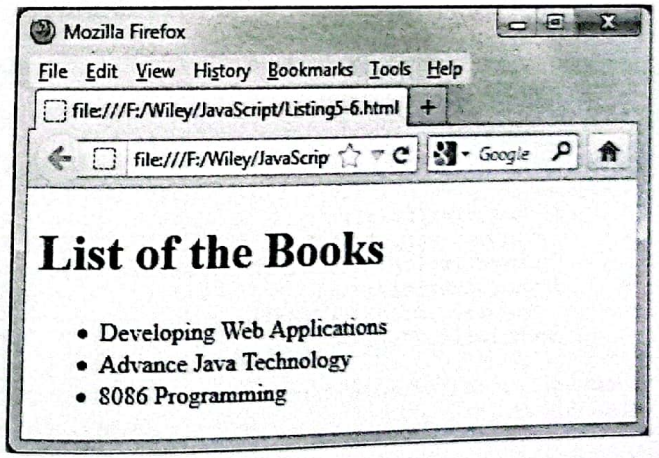


Figure 4 Output of Listing 6.

```
<html>
<body>
<script type="text/javascript">
   var books = new Array("Web Applications", "Java", "8086 Programmng");
   document.write("<h1>List of the Books</h1>");
   document.write("<ul>");
   var i;
   for(i=0;i<books.length;i++) {
      document.write("<li>");
      document.write(books[i]);
      document.write("</li>");
   }
   document.write("</ul>");
</script>
</body>
</html>
```

Listing 7 Display list of books.

Listing 8 receives an integer value N (> 0) from the user and displays the table of squares for numbers 0 to $N - 1$ where the first column is the number itself and the second column is the square of the number. Remember that the table content is dynamically generated and also size of the table depends on the user input. Thus, the resulting Web page is generated totally dynamically.

```
<html>
<body>
<script type="text/javascript">
      var n = prompt("Enter value of N");
      document.write("<h1>Diplaying Table of Squares</h1>");
      document.write("<table border=\"1\">");
      document.write("<tr>");
      document.write("<th>Number</th>");
      document.write("<th>Square</th>");
      document.write("</tr>");
      var i;
      for(i=0;i<n;i++) {
         document.write("<tr>");
            document.write("<td align=\"center\">");
            document.write(i);
            document.write("</td>");
            document.write("<td align=\"center\">");
            document.write(i*i);
              document.write("</td>");
         document.write("</tr>");
      }
      document.write("</table>");
</script>
</body>
</html>
```

Listing 8 Generating Web page with table of squares.

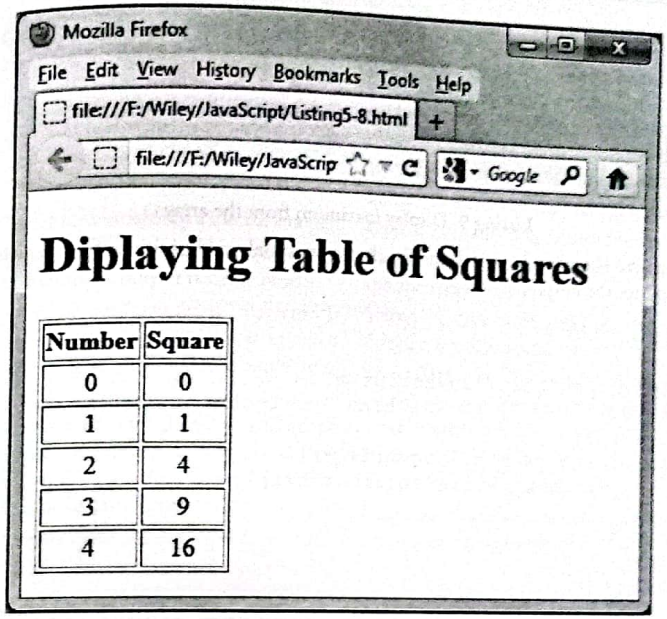


Figure 5 Web page generated by Listing 8.

The executing code in Listing 8 shows the prompt box to get the value of N. If the value entered in prompt box is 5 and then "OK" button is pressed, it produces output as shown in Fig. 5.

Listing 9 finds the maximum value from the given array using function and displays it as h1. Run it in browser and see the output.

```
<html>
<head>
<script type="text/javascript">
      function max(a, b)
      {
         return a>b?a:b;
      }
</script>
</head>
<body>
<script type="text/javascript">
      var a = new Array(10,7,23,17,14);
      var max_val = a[0];
      for(i=1;i<a.length;i++) {
         max_val = max(max_val, a[i]);
      }
```

```
        document.write("<h1>The largets is :");
        document.write(max_val);
        document.write("</h1>");
    </script>
    </body>
    </html>
```

Listing 9 Display maximum from the array.

Listing 10 prints the alternate characters, that is, characters at only odd positions from a given string. Run it in browser and see the output.

```
    <html>
    <body>
    <script type="text/javascript">
        var str = "I am learning Web technologies";
        var i;
        for(i=1;i<=str.length;i+=2){
           document.write(str.charAt(i));
        }
    </script>
    </body>
    </html>
```

Listing 10 Display alternate characters from a string.

The above examples provided us the idea that how JavaScript is used to make the Web pages dynamic.

Checkpoint!

JavaScript makes Web pages interactive using popup boxes which includes alert, confirm and prompt boxes.

- HTML provides only static contents which do not change. The JavaScript adds dynamism to the Web pages by executing script code at client side.

Summary

1. Client side scripting has some benefits over static Web pages and server side scripting too.
2. JavaScript has been around since the early days of the Web.
3. JavaScript allows pages to become dynamic and interactive.
4. It has a similar language structure to C, C++ and Java, although there are some interesting differences.
5. JavaScript can be developed using just a simple editor and browser or a Web development IDE.
6. JavaScript can be embedded in a Web page or linked in as an external file.
7. Variables are untyped and do not have a block-scope as in some languages.
8. Strings and arrays are both objects in JavaScript, which have built-in properties and methods.
9. Functions are sections of code that can be invoked with parameters and return a value. They are also classed as an actual value type.
10. There are several language features that allow program flow to be changed on condition, or repeated.

Key Words and Phrases

Array In JavaScript, an object that contains data and has properties and useful methods such as `pop`, `push` and `join`. The data can be of mixed type.

Assignment When variables take on a value.

Decrement To reduce a variable by a specific amount.

Dynamic Web page A page that reacts to events or user input and alters its output because of it.

Escape sequence When a special character is included within a string, a backslash is required before it to enable it to be output. An example is the single quote.

Function A section of named code that can be invoked many times during a program, can also be sent parameters and have a return value. A function is also a variable type in JavaScript.

Increment To increase a variable by a specific amount.

Invoke To start, or call, a function to enable it to execute.

Iterate To repeat a section of code.

Loop A section of code that is repeated.

Object A collection of data and methods.

Scope The containment of a variable within a specific area of code.

Script Usually a program that is interpreted by a browser, e.g. JavaScript.

Static Web page A page that simply formats its output without change or reaction to the user or other information.

String An object that is a collection of characters wrapped by quotes and also has properties and methods.

Untyped Variables are untyped when they do not declare their type before use.

Multiple-Choice Questions

1. What is the advantage of server side scripting?
   (a) Reduces the burden on server
   (b) Code is visible
   (c) Code is modifiable
   (d) All of the above
2. What was original name of the JavaScript?
   (a) ECMAScript
   (b) JScript
   (c) LiveScript
   (d) None of the above
3. Which of the following is true about JavaScript?
   (a) It is client side scripting language
   (b) It is object-based language
   (c) It is used to make Web interactive
   (d) All of the above
4. How JavaScript code is embedded in HTML?
   (a) `<script>`
   (b) `<javascript>`
   (c) `<js>`
   (d) None of the above
5. If a JavaScript variable is declared without reserved word "`var`", then it is
   (a) Local if inside the function
   (b) Global if outside the function
   (c) Always global
   (d) None of the above
6. Which of the following is JavaScript concatenation operator?
   (a) #
   (b) +
   (c) .
   (d) $
7. Which of the following is true about JavaScript function?
   (a) Itself is data type
   (b) Passed to other function
   (c) Stored in variable
   (d) All of the above
8. Which of the following looping structure is not supported by JavaScript?
   (a) for
   (b) for-each
   (c) while
   (d) do-while

CamScanner

Review Questions

1. What client side scripting? Give its advantages and disadvantages.
2. List the characteristics of JavaScript.
3. Explain using examples how to write embedded and external JavaScripts.
4. Differentiate between local and global variables in JavaScript with example.
5. What is a function in JavaScript? Give an example.
6. Explain the string processing in JavaScript.

Practice

1. Write a JavaScript program to print your name and address in proper format.
2. Write a JavaScript program to find minimum of three numbers.
3. Write a JavaScript program to display your last semester mark sheet in tabular format. Use HTML and CSS for structure and styles, respectively.
4. Write a JavaScript program to display first *N* Fibonacci numbers.
5. Write a JavaScript program to sort the array of *N* elements.
6. Write a JavaScript program to read an integer number *N* from user and display all the numbers between 1 and *N* that are divisible by 9.
7. Study more about string processing functions and other built-in functions of JavaScript from various Web resources.
8. Develop a tutorial Web site to teach arithmetic operations to primary school students. Make the use of HTML, CSS and JavaScript to make your Web site rich in all aspects.

Chapter Quiz

1. What do you think JavaScript is particularly useful for, that HTML isn't?
2. Try and find some good reference sources on the Web for JavaScript; you can start with the ones below.
3. Do you think JavaScript is object oriented? Try and find out why some authors call it object based – why do they make the distinction?

Useful Web Address

http://www.ecma-international.org/publications/standards/Stnindex.htm

Answers

1. (a)
2. (c)
3. (d)
4. (a)
5. (c)
6. (b)
7. (d)
8. (b)

CamScanner