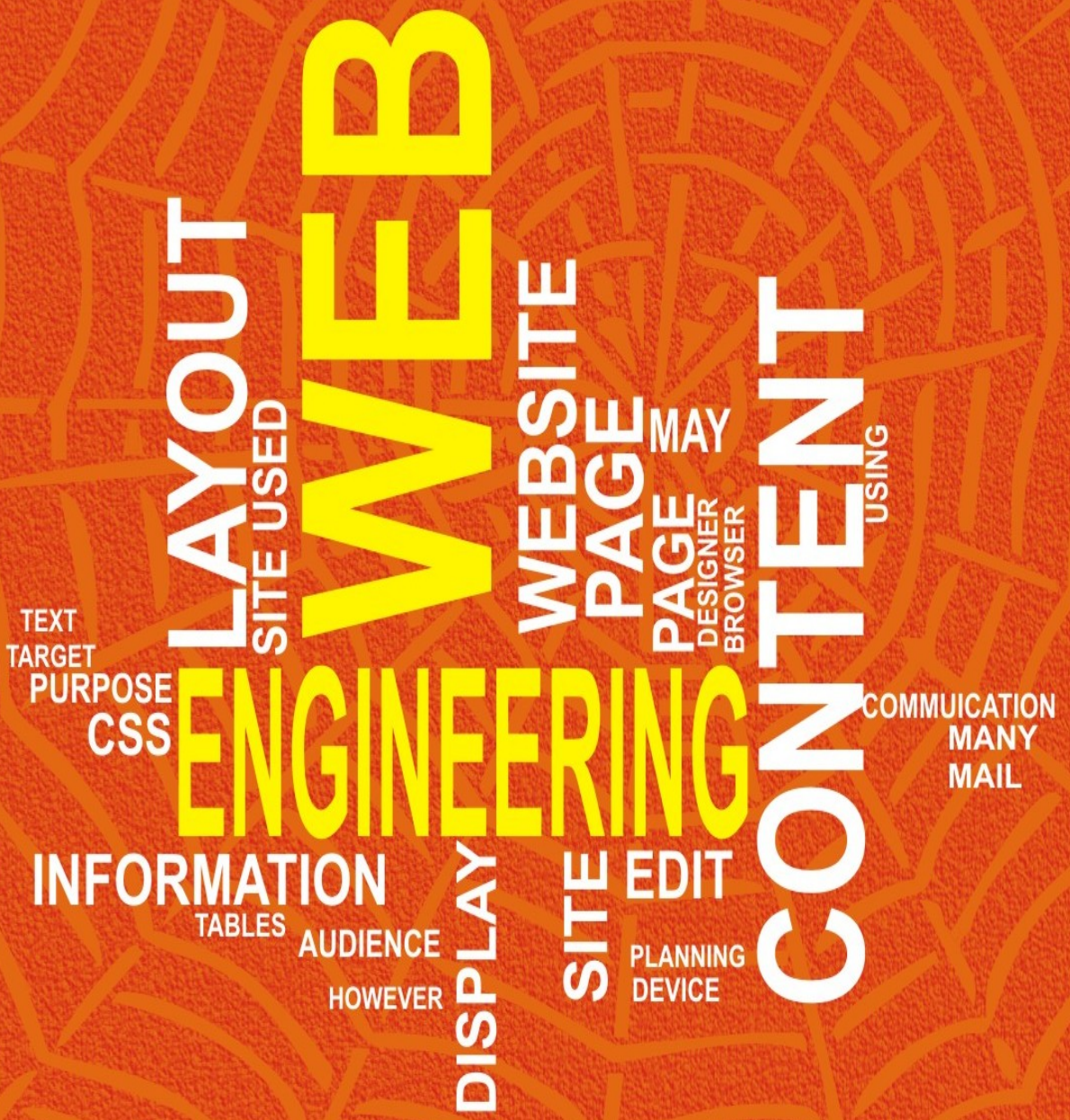


Pankaj Kapoor

WEB ENGINEERING

FOR

[B. Tech., M.Sc., M.C.A., M.Tech.]

Pankaj Kapoor
(Assistant Professor)
M.Tech., Kurukshetra University

Soniya Kapoor
(Assistant Professor)
M.Tech., J.M.I.T., Raduar

BHARAT PUBLICATIONS

5A/12, Ground Floor, Ansari Road, Darya Ganj, New Delhi - 110002
B.O. 135-A, Santpura Road, Yamuna Nagar - 135001 (Haryana)
Phone: 01732-227178, 232178, 09416227140
Email: bharatpublications.ynr@gmail.com

Dedicated to our Beloved Parents

ISBN– 13:978–93–81252–29–1

First Edition:July 2012

Price : 680.00

Publishing by : Bharat Publications

Type Set :
Kamal Computers : 96710-46578 (Y.N.R.)

PREFACE

We have great pleasure in presenting the First edition of this book. **"Web Engineering"** is a core subject for undergraduate students and Post-graduate students in Computer Science Engineering (CSE), Information Technology Streams and Master of Computer Application (MCA). This book is primarily intended to serve as a textbook in accordance with syllabus of Introduction to Web Engineering And Technology offered by various universities in India as well as abroad.

The authors have identified the problems of the subject at students' level. In this book, a significant effort has been made to find simple ways to develop theoretical as well as Practical concept of Web Engineering. A very strong emphasis has been given on Practical examples. Neat and Clean diagrams have been used for explanation. Every chapter has been divided and sub-divided into useful learning points and adequately discussed. Even minute problems have been thoroughly discussed in a lucid manner. The discussions have been supported by suitable illustrations and diagrams.

. The book reflects the knowledge of the subject and style of presentation of authors. This book is divided into ten chapters:

The organizations of these chapters are:

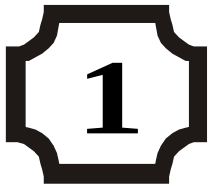
Chapter 1 Introduces the concepts of Web Engineering.
Chapter 2 Introduces the World Wide Web.
Chapter 3 Introduces the Information Architecture.
Chapter 4 Explain the concept of Creating Cohesive Websites.
Chapter 5 Introduces the HTML (Hypertext Markup languages).
Chapter 6 Introduces the concepts of Web Designing.
Chapter 7 Introduces the concepts of CGI.
Chapter 8 Introduces the concepts of CGI Environment using Perl.
Chapter 9 Introduces the concepts of JSP.
Chapter 10 Introduces the concepts of XML.

We have made our sincere efforts to make the book meaningful comprehensive, compact and complete but even then there is always room for improvement. Suggestions for the improvement of the book will be gratefully acknowledged.

It will be a pleasure to share your experiences with the book. We shall gratefully acknowledge the valuable comments and suggestions sent in by the dear readers, which I have earnestly tried to act upon while shaping the present edition.

Authors

Contents



INTRODUCTION TO WEB ENGINEERING

IN THIS CHAPTER, YOU WILL LEARN

- History of the Web Development
- Introduction to Web Engineering
- Web Engineering as a Discipline
- Web Engineering Activities
- Web Applications
- Web-Based Systems Development
- Web Application Testing
- Web characteristics
- Evolution and Need For Web Engineering
- Web Engineering and Web Gardening
- Web Engineering Skills.

The World Wide Web has become a major delivery platform for a variety of complex and sophisticated enterprise applications in several domains. In addition to their inherent multifaceted functionality, these web applications exhibit complex behavior and place some unique demands on their usability, performance, security and ability to grow and evolve. However, a vast majority of these applications continue to be developed in an ad-hoc way, contributing to problems of usability, maintainability, quality and reliability. While web development can benefit from established practices from other related disciplines, it has certain distinguishing characteristics that demand special considerations. In the recent years, there have been some developments towards addressing these problems and requirements. As an emerging discipline, web engineering actively promotes systematic, disciplined and quantifiable approaches towards successful development of high-quality, ubiquitously usable web-based systems and applications.

1.1 HISTORY OF WEB DEVELOPMENT

Since the mid-1990s, web development has been one of the fastest growing industries in the world. In 1995 there were fewer than 1,000 web development companies in the United States, but by 2005 there were over 30,000 such companies in the U.S. alone. The growth of this industry is being pushed by large businesses wishing to sell products and services to their customers and to automate business workflow. Web development is a broad term for the work involved in developing a web site for the Internet (World Wide Web) or an intranet (a private network). This can include web design, web content development, client liaison, client-side/server-side scripting, web server and network security configuration, and e-commerce development. However, among web professionals, "web development" usually refers to the main non-design aspects of building web sites: writing markup and coding. Web development can range from developing the simplest

static single page of plain text to the most complex web-based internet applications, electronic businesses, or social network services.

For larger organizations and businesses, web development teams can consist of hundreds of people (web developers). Smaller organizations may only require a single permanent or contracting webmaster, or secondary assignment to related job positions such as a graphic designer and/or information systems technician. Web development may be a collaborative effort between departments rather than the domain of a designated department.

Along with the development of desktop software for web developers, a movement to create web sites without having to download any desktop software had begun. The theory was that users could use an Internet browser such as Netscape, Firefox, Safari and / or Internet Explorer to login and then make changes to their web site. The solutions were built using dynamic pages as described above, and also connected to a database. These solutions were called Content Management Systems (CMS).

A Web Content Management System (WCMS) is a software system that provides website authoring, collaboration, and administration tools designed to allow users with little knowledge of web programming languages or markup languages to create and manage website content with relative ease. A robust WCMS provides the foundation for collaboration, offering users the ability to manage documents and output for multiple author editing and participation.

- Most systems use a Content Repository or a database to store page content, metadata, and other information assets that might be needed by the system.
- A presentation layer displays the content to website visitors based on a set of templates. The templates are sometimes XSLT files.
- Most systems use server side caching to improve performance. This works best when the WCMS is not changed often but visits happen regularly.
- Administration is typically done through browser-based interfaces, but some systems require the use of a fat client.
- A WCMS allows non-technical users to make changes to a website with little training. A WCMS typically requires a systems administrator and/or a web developer to set up and add features, but it is primarily a website *maintenance* tool for non-technical staff.

1.2 INTRODUCTION TO WEB ENGINEERING

In particular, web engineering focuses on the methodologies, techniques and tools that are the foundation of web application development and which support their design, development, evolution, and evaluation. Web application development has certain characteristics that make it different from traditional software, information system, or computer application development. Web engineering is multidisciplinary and encompasses contributions from diverse areas: systems analysis and design, software engineering, hypermedia/hypertext engineering, requirements engineering, human-computer interaction, user interface, information engineering, information indexing and retrieval, testing, modeling and simulation, project management, and graphic design and presentation. Web engineering is neither a clone, nor a subset of software engineering, although both involve programming and software development.

While web Engineering uses software engineering principles, it encompasses new approaches, methodologies, tools, techniques, and guidelines to meet the unique requirements of web-based applications.

Engineering in general means the practical application of science to commerce or industry with the goal of designing applications in a better, faster, cheaper and in more secure way.

Software Engineering is defined as the application of science and mathematics by which the capabilities of computer equipment are made useful to man via computer programs, procedures, and associated documentation. Based on this definition we can defined *Web Engineering* as follows:

1. Web Engineering is the application of systematic and quantifiable approaches (concepts, methods, techniques, tools) to cost-effective requirements analysis, design, implementation, testing, operation, and maintenance of high-quality Web applications.
2. Web Engineering is also the scientific discipline concerned with the study of these approaches.

From the point of view of Software Engineering, the development of Web applications is a new application domain. Despite some similarities to traditional applications, the special characteristics of Web applications require an adaptation of many Software Engineering approaches or even the development of completely new approaches.

The basic principles of Web Engineering can, however, be described similarly to those of Software Engineering:

- Clearly defined goals and requirements
- Systematic development of a Web application in phases
- Careful planning of these phases
- Continuous audit of the entire development process.

Web Engineering makes it possible to plan and iterate development processes and thus also facilitates the continuous evolution of Web applications. This permits not only cost reduction and risk minimization during development and maintenance, but also an increase in quality, as well as measurement of the quality of the results of each phase.

1.3 WEB ENGINEERING AS A DISCIPLINE

Proponents of web engineering supported the establishment of web engineering as a discipline at an early stage of web. First Workshop on Web Engineering was held in conjunction with World Wide Web Conference held in Brisbane, Australia, in 1998. San Murugesan, Yogesh Deshpande, Steve Hansen and Athula Ginige, from University of Western Sydney, Australia formally promoted web engineering a new discipline in the first ICSE workshop on Web Engineering in 1999. Since then they published a serial of papers in a number of journals, conferences and magazines to promote their view and got wide support. Major arguments for web engineering as a new discipline are:

- Web-based Information Systems (WIS) development process is different and unique.
- Web engineering is multi-disciplinary; no single discipline (such as software engineering) can provide complete theory basis, body of knowledge and practices to guide WIS development.

- Issues of evolution and lifecycle management when compared to more 'traditional' applications.
- Web based information systems and applications are pervasive and non-trivial. The prospect of web as a platform will continue to grow and it is worth being treated specifically.

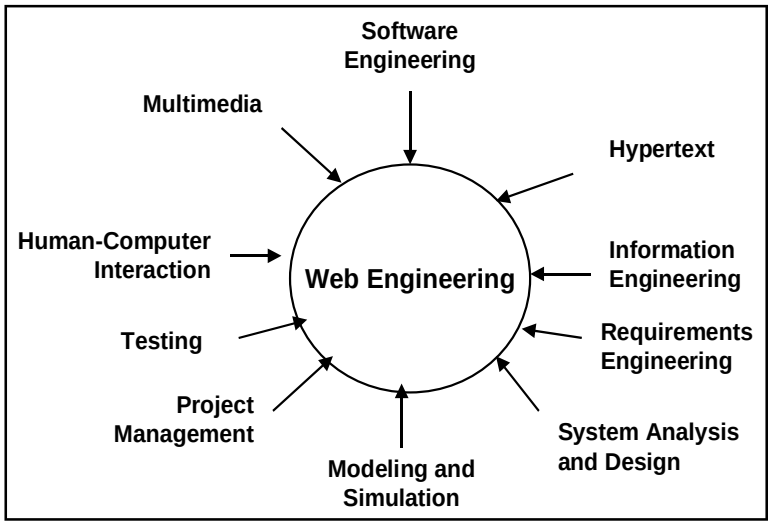


Fig. 1.1 : Web Engineering – A multidisciplinary field.

However, it has been controversial, especially for people in other traditional disciplines such as software engineering, to recognize web engineering as a new field. The issue is how different and independent web engineering is, compared with other disciplines.

Main topics of Web engineering include, but are not limited to, the following areas:

Web Requirements Modeling Disciplines :

- Business Processes for Applications on the Web
- Process Modeling of Web applications
- Requirements Engineering for Web applications
- Web System Design Disciplines, Tools & Methods
- UML and the Web
- Conceptual Modeling of Web Applications (aka. Web modeling)
- Prototyping Methods and Tools
- Web design methods
- CASE Tools for Web Applications
- Web Interface Design
- Data Models for Web Information Systems
- Web System Implementation Disciplines
- Integrated Web Application Development Environments
- Code Generation for Web Applications
- Software Factories for/on the Web
- Web 2.0, AJAX, E4X, ASP.NET, PHP and Other New Developments

- Web Services Development and Deployment
- Empirical Web Engineering
- Web System Testing Disciplines
- Testing and Evaluation of Web systems and Applications
- Testing Automation, Methods and Tools

Web Applications Categories Disciplines :

- Semantic Web applications
- Ubiquitous and Mobile Web Applications
- Mobile Web Application Development
- Device Independent Web Delivery
- Localization and Internationalization Of Web Applications

Web Quality Attributes Disciplines :

- Web Metrics, Cost Estimation, and Measurement
- Personalization and Adaptation of Web applications
- Web Quality
- Usability of Web Applications
- Web accessibility
- Performance of Web-based applications

Content-related Disciplines :

- Web Content Management
- Multimedia Authoring Tools and Software
- Authoring of adaptive hypermedia

1.4 WEB ENGINEERING ACTIVITIES

Web Engineering is not a single activity or task. It deals with all aspects of Web-based system development, starting from conception and development to implementation, performance evaluation, and continual maintenance. Major Web engineering include:

- Requirements specification and analysis
- Web-based system development methodologies and techniques
- Integration with legacy systems
- Migration of legacy system to Web environments
- Web-based real-time applications development
- Testing, verification and validation
- Quality assessment, control and assurance
- Configuration and project management
- “Web metrics” - metrics for estimation of development efforts
- Performance specification and evaluation
- Update and maintenance

- Development models, teams, staffing
- Human and cultural aspects
- User-centric development, user modeling and user involvement and feedback
- End-user application development
- Education and training

1.5 WEB APPLICATIONS

A Web application is defined as any application program that runs on the Internet or corporate intranets and extranets. The user of a Web application uses a Web browser on a client computer to run the program residing on the server. The entire processing is done on the server as if it were done at the user's local machine. In this chapter we use the term in a broader context to include any application that is Web browser based.

1.5.1 History of Web Application

In earlier computing models, e.g. in client-server, the load for the application was shared between code on the server and code installed on each client locally. In other words, an application had its own client program which served as its user interface and had to be separately installed on each user's personal computer. An upgrade to the server-side code of the application would typically also require an upgrade to the client-side code installed on each user workstation, adding to the support cost and decreasing productivity.

In contrast, web applications use web documents written in a standard format such as HTML and JavaScript, which are supported by a variety of web browsers. Web applications can be considered as a specific variant of client-server software where the client software is downloaded to the client machine when visiting the relevant web page, using standard procedures such as Http. Client web software update may happen each time the web page is visited. During the session, the web browser interprets and displays the pages, and acts as the *universal* client for any web application. In the early days of the Web each individual web page was delivered to the client as a static document, but the sequence of pages could provide an interactive experience, as user input is returned through web form elements embedded in the page markup.

In 1995 Netscape introduced a client-side scripting language alled JavaScript allowing programmers to add some dynamic elements to the user interface that ran on the client side. So instead of sending data to the server in order to generate an entire web page, the embedded scripts of the downloaded page can perform various tasks such as input validation or showing/hiding parts of the page. In 1996, Macromedia introduced Flash, a vector animation player that could be added to browsers as a plug-in to embed animations on the web pages. It allowed the use of a scripting language to program interactions on the client side with no need to communicate with the server.

In 1999, the "web application" concept was introduced in the Java language

in the Servlet Specification version 2.2. At that time the Java Script and XML had already been developed, but Ajax had still not yet been coined and the XML HttpRequest object had only been recently introduced on Internet Explorer 5 as an ActiveX object.

In 2005, the term Ajax was coined, and applications like Gmail started to make their client sides more and more interactive. A web page script is able to contact the server for storing/retrieving data without downloading an entire web page.

In 2011, HTML5 was created, which provides graphic and multimedia capabilities without the need of client side plugins. HTML5 also enriched the semantic content of documents. The APIs and document object model (DOM) are no longer afterthoughts, but are fundamental parts of the HTML5 specification. WebGL API paved the way for advanced 3D graphics based on HTML5 canvas and JavaScript language. These have significant importance in creating truly platform and browser independent rich web applications.

1.5.2 Categories of Web Applications

Web applications have varying degrees of complexity. They may be purely informational or handle full-size/full-fledged 24/7 e-commerce applications. Web applications depending upon their development history and their degree of complexity. We must bear in mind that there is a correlation between the chronology of development and complexity. The development of a Web application can be started in any of these categories and later expanded to increasing degrees of complexity. Newer categories are generally more complex, but this does not mean they can fully replace the older generation. Each of these categories has its own specific fields of application. In consequence, complex Web applications in particular can typically be assigned to several categories at once. We will now describe the relevant features of these categories:

a) Document centric: Document centric Web sites are the precursor to Web applications. Web pages are stored on a Web server as ready-made, i.e. static, HTML documents and sent to the Web client in response to a request. These Web pages are usually updated manually using respective tools. The main benefits are the simplicity and stability of such Web sites and the short response time, as the pages are already stored on the Web server. Static homepages, webcasts, and simple web presences for small businesses belong in this category.

b) Interactive Web applications: With the introduction of the Common Gateway Interface and HTML forms, interactive Web applications emerged, offering a first, simple, form of interactivity by means of forms, radio buttons and selection menus. Web pages and links to other pages are generated dynamically according to user input. Examples for this category are virtual exhibitions, news sites, or timetable information.

c) Transactional Web applications: Transactional Web applications were created to provide more interactivity, giving the user the possibility of not only interacting with the application in a read-only manner, but also by performing updates on the underlying content. Online banking, online shopping, and booking systems belong in this category.

d) Work Flow-based : Work flow-based Web applications allow the handling of workflows within or between different companies, public authorities, and private users. A driving force for this is the availability of appropriate Web services to guarantee interoperability (Weerawarana et al. 2005). The complexity of the services in question, the autonomy of the participating companies and the necessity for the workflows to be robust and flexible are the main challenges. Examples for this category are Business-to-Business solutions (B2B solutions) in e-commerce, e-government applications in the area of public administration, or Web-based support of patient work flows in the health sector.

e) Collaborative Web applications: Collaborative Web applications are employed especially for cooperation purpose in unstructured operations (groupware). There the need for communication between the cooperating users is particularly high. Collaborative Web applications support shared information and workspaces (e.g. Wikipedia) in order to generate, edit, and manage shared information. They are also used to keep logs of many small entries and edits (as in Weblogs), to mediate meetings or make decisions or simple chat rooms), as scheduling systems, or as e-learning platforms.

f) Portal-oriented Web applications: Portal-oriented Web applications provide a single point of access to separate, potentially heterogeneous sources of information and services (Wege 2002). Makers of browsers, such as Microsoft and Netscape, search engines such as Yahoo, online services such as AOL, media conglomerates, and other companies have become aware of the demand for this and now offer central hubs, so-called portals, as a point of access to the Web. In addition to these general portals, there are various specialized portals such as business portals, marketplace portals in the form of online shopping malls, and community portals.

g) Ubiquitous Web applications: Ubiquitous Web applications provides customized services anytime anywhere and for any device, thus facilitating ubiquitous access. An example of this would be displaying the menu of the day on the mobile devices of all users entering a restaurant between 11 am and 2 pm. For this type of system it is important to take into account the limitations of mobile devices (bandwidth, screen size, memory, immaturity of software, etc.) and the context in which the Web application is currently being used.

h) Semantic Web: The goal of the Semantic Web is to present information on the Web not merely for humans, but also in a machine-readable form. This would facilitate knowledge management on the Web, in particular the linking and reuse of knowledge (content syndication), as well as locating new relevant knowledge, e.g. by means of recommender systems.

1.5.3 Characteristics of Web Applications

Web applications differ from traditional, non-Web-based applications in a variety of features worth looking into. These are characteristics that traditional applications lack completely (e.g. non-linear navigation) on the one hand and characteristics that are of particular importance in Web applications on the other hand (e.g. frequency of updates). Whether a certain characteristic is present and to what degree depends

partly on the type of Web application: the development of transactional Web applications such as e-commerce systems requires greater focus on the content being up to date and consistent as compared with pure information provision systems – e.g. virtual exhibitions. These characteristics are the reason why many concepts, methods, techniques, and tools of traditional Software Engineering have to be adapted to the needs of Web Engineering or may even be totally inadequate. Fig. 1-2 gives an overview of these characteristics and arranges them along the three dimensions: "product", "usage", and "development" with their "evolution" as an encompassing dimension.

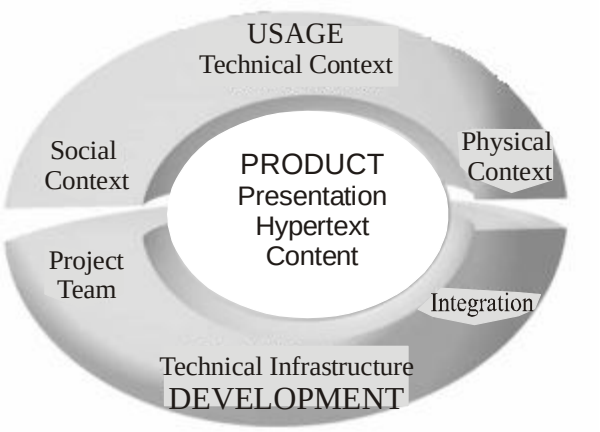


Fig. 1.2

A) Product-related Characteristics : Product-related characteristics constitute the major building blocks of a Web application, consisting of content, the hypertextual structure (navigational structure), and presentation (the user interface). Following the object-oriented paradigm, each of these parts has not only a structural or static aspect, but also a behavioral or dynamic aspect.

1. Content : Generating content, making it available, integrating, and updating it is equally important as the development and provision of the actual software of a Web application. Web applications are used expressly because of the content Web application developers must therefore not only act as programmers but also as authors. Important aspects are the varying degree of structure of the content and the quality demands users make on the content.

2. Hypertext : Amongst the specific characteristics of Web applications is the non-linear nature of hypertextual documents. The hypertext paradigm as a basis for the structuring and presentation of information was first mentioned by Vannevar Bush (Bush 1945). Basic elements of hypertext models are nodes, links and anchors. A node is a self-contained uniquely identifiable information unit. On the Web this might be an HTML document which can be reached via a URL (Uniform Resource Locator). A link is the path from one node to another. On the Web, these paths are always unidirectional and their meaning is not clearly defined. Possible meanings include "next node according to recommended reading order" or "diagram for mathematical formula". An anchor is the area within the content of a node that is the source or destination of a link, e.g. a sequence of words in a text or a graphical object in a drawing. On the Web, anchors are only possible in HTML documents.

3. Presentation : Two special features of Web applications at the presentation level, i.e. the user interface, are aesthetics and self-explanation.

- **Aesthetics:** In contrast to traditional applications, the aesthetics of the presentation level of a Web application, the "look and feel" of the user

interface, is a central factor not least because of the high competitive pressure on the Web. The visual presentation of Web pages is subject to fashion trends and often determines success or failure, in particular for e-commerce applications.

- **Self-explanation:** Besides aesthetics, it is essential that Web applications are self- explanatory, i.e. it should be possible to use a Web application without documentation. The navigation system or interaction behavior must be consistent within the whole application, so that users can quickly become familiar with the usage of the Web application.

B) Usage-related Characteristics : Compared with traditional applications, the usage of Web applications is extremely heterogeneous. Users vary in numbers and cultural background, devices have differing hardware and software characteristics, and the time and location from where the application is accessed cannot be predicted. The usage of Web applications is therefore characterized by the necessity to continuously adapt to specific usage situations, so-called contexts. Adjustment to these contexts can be equally necessary for all parts of the Web application, i.e. content, hypertext, and presentation. Because of the fundamental significance of adjustment to contexts, usage-related characteristics are divided into three groups : social context, technical context, and natural context.

1. **Social Context: Users :** The social context refers to user-specific aspects; spontaneity and multiculturality in particular create a high degree of heterogeneity.
2. **Technical Context: Network and Devices :** The technical context comprises properties relating to the network connection concerning quality of service, and the hardware and software of the devices used to access the Web application, for multi-platform delivery.
3. **Natural Context: Location and Time :** The natural context includes aspects of the location and time of access. Globality and availability create a high degree of heterogeneity.

C) Development-related Characteristics : The development of Web applications is characterized by the necessary resources, such as the development team and the technical infrastructure, the development process itself, and the necessary integration of already existing solutions.

The Development Team : The development of Web applications is strongly influenced by the fact that development teams are multidisciplinary and generally rather young. These factors and the methods of the so-called community development contribute to a completely new way of organizing collaboration of different groups of developers.

1. **Technical Infrastructure :** The in homogeneity and immaturity of the used components are important characteristics of the technical infrastructure of Web applications.
2. **Process :** The development process is the framework for all development-related characteristics, and is in turn influenced by flexibility and parallelism.
3. **Integration :** A special characteristic of many Web applications is the need

for internal and external integration. Integration in this context refers not only to technical aspects, but also to content, and organizational aspects.

Advantages : Web based applications promise a number of advantages over traditional non-Web based applications.

1. Web applications do not require any complex "roll out" procedure to deploy in large organizations. A compatible web browser is all that is needed;
2. Browser applications typically require little or no disk space on the client;
3. They require no upgrade procedure since all new features are implemented on the server and automatically delivered to the users;
4. Web applications integrate easily into other server-side web procedures, such as email and searching.
5. They also provide cross-platform compatibility in most cases (i.e., Windows, Mac, Linux, etc.) because they operate within a web browser window.
6. With the advent of HTML5, programmers can create richly interactive environments natively within browsers. Included in the list of new features are native audio, video and animations, as well as improved error handling.

Disadvantages : Web applications are not the silver bullet that everyone has been dreaming about for so long. Depending on how a Web application is built and what technologies are chosen, some things must be given up.

1. **Speed Loss:** Web applications do not run as fast as those running on local machine because of the downloading time and network traffic. This may become less a problem as computer hardware and software improve.
2. **Data Presentation Limit:** If you choose to go with server-side Javascripting or a total HTML solution, such as is available via a tool like Intrabuilder, you may be limited to the interface defined by HTML. In other words, you may be unable to provide the users with the latest in the widgets and gadgets that the modern user interface can provide. For example, tools such as data grids and their capabilities are currently not available. This may limit your ability to layout clearly an application and present data to the user. However, coming advances in HTML technology will reduce this limitation, as the HTML interface becomes more sophisticated.
3. **Security Vulnerability:** Web applications are inherently vulnerable to malicious Internet attacks. These attacks can be classified as vandalism and sabotage, breach of privacy, theft and fraud, violations of data integrity, and denial of service. As the e-commerce technologies become more sophisticated, these threats will be minimized.

1.6 WEB-BASED SYSTEMS DEVELOPMENT

As highlighted in the previous sections, Web engineering activities span the entire Web life cycle from conception of an application to development and deployment, and continual refinement and update/upgrade systems. The following

highlights some of the work and development in the area of Web engineering. They are however, not an extensive survey or critical review of the work reported.

1. Web Development Process Models : To help to reduce the difficulty in building Web-based systems we need a process model that describe the phases of Web-based system development some of the aspects that make Web-system difficult include complexity, changeability, invisibility and unrealistic schedule. A process model should help developers "to address the complexities of Web-based systems, minimize risks of development, deal with likelihood of change, and deliver the site quickly, while providing feedback for management as the project goes along." Further, the progress of Web-based development should be monitorable and trackable. The process besides being easy to apply should facilitate continual update/refinement and evolution, based on feedback from users/clients. For information some of the hypermedia/Web development process models. An object-oriented model for the Web application development process, which uses XML technology to support modularity and reuse of Web document.

2. Analysis and Web Design : Requirement analysis and Web-based system design is a very important activity and calls for a systematic and disciplined approach.

- **Object Orientation in Web-Based Systems :** Integration of Web and object technologies offer foundation for expanding the Web to a new generation of applications. According to Frank Manolo, Web must improve its data structuring capabilities, and integrate aspects of object technology with the basis infrastructure of the Web. He also argues that if the Web is to support complex enterprise applications, it must support generic capabilities similar to those provided by the OMA (Object management Architecture), but adapted to the more open, flexible nature of the Web and to the specific requirements of Web applications.

- **Usability and User-Centered Designs :** Effective Web site design requires attention to usability. Web-based systems need to be designed for easy navigation, and also they need to be attractive and useful. User-centered design methods for Web sites is presented in, while presents a User-Centric Approach to Modeling Web Information Systems.

3. Testing of Web-Based Systems : Testing, and verification and validation (V & V) of Web-based systems is an important and challenging task in the Web engineering process. And, yet very little attention is given by Web developers to testing and evaluation. Web-based system testing differs from conventional software testing and poses new challenges. Web-based systems need to be tested not only to check and verify whether it does what it is designed to do but also to evaluate how well it appears on (different) Web browsers. Importantly, they need to be tested for security and also for usability, from the ultimate user's perspective. However, the unpredictability of the Internet and Web medium makes testing Web based systems difficulty. Currently, not much attention is given to Web–based system testing by developers. Also we need to develop new approaches and techniques for testing and evaluation of complex Web-based systems.

4. Management of Large Web Sites : Management of large Web sites is a difficult task, especially in the midst of change which is a fact of life in the Web environment.

Requirements for management of large Web sites and the tools and a mechanism for organizing and manipulating large Web sites..

- **Web Configuration Management :** Web-based systems undergo changes, perhaps more often and quite extensively, in their development and operational period. The changes called for may include trivial to large-scale change of information/ data and major modification to requirements, and also may vary in their significance. These changes need to be handled in a rational, controlled manner. Web configuration management (WCM) encompasses a set of activities for controlling and facilitating change: identification, version control, change control, auditing and reporting. It also provides a framework for handling change in a rational, controlled manner. It could adopt commonly practiced software configuration management (SCM) concepts, principles and approaches to the Web environment.

5. Skills Hierarchy : Large Web-based system development requires a team of people with different skills, knowledge and capabilities. A categorization of skills and knowledge-base hierarchy for participants in Web-based system development is provided.

6. Barriers to Web Technology Adoption :Three levels of adoption of Web technology: (Level 1) information access, (level 2) work collaboration, and (Level 3) core business transaction. They also identify three key of potential knowledge barriers to Web technology adoption: technology-related knowledge barriers, project related knowledge barriers, application related knowledge barriers.

1.7 WEB APPLICATION TESTING

Web applications pose new challenges to quality assurance and testing. Web applications consist of diverse software components possibly supplied by different manufacturers. The quality of a Web application is essentially determined by the quality of each software component involved and the quality of their interrelations. Testing is one of the most important instruments in the development of Web applications to achieve high-quality products that meet users' expectations.

Testing is an activity conducted to evaluate the quality of a product and to improve it by identifying defects and problems. If we run a program with the intent to find errors, then we talk about testing. We say that an error is present if the actual result from a test run does not comply with the expected result. Testing won't lead to quality improvement unless errors are detected and removed. The main test objective is to find errors, rather than to show their absence. Software tests are unsuitable to prove the absence of errors. If a test doesn't find errors, then this does not mean that the tested application doesn't contain any. They may simply not have been detected yet.

When testing Web applications, we can basically apply all methods and techniques commonly used in traditional software testing. To take the specifics of Web applications into account, some of these test methods and techniques will have to be thought over, or adapted and expanded (e.g., "What influence factors have to be taken into account when testing compatibility with different Web browsers?").

1. Functional Testing : Functional testing involves making sure that the features that most affect user interactions work properly. In this the following should be tested:

- **Forms :**
  (i) Field validation
  (ii) Error message for wrong input
  (iii) Optional and Mandatory fields
- **Database :** Testing will be done on the database integrity.
- **Cookies :** Testing will be done on the client system side, on the temporary Internet files.
- Pop-up windows
- Shopping Carts
- Online Payments

2. Navigation Testing : Links within a hypertext navigation structure that point to a non-existing node (pages, images, etc.) or anchor are called broken links and represent well-known and frequently occurring errors in Web applications. To test for correct linking of pages (link checking), all links are systematically followed beginning on a start page, and then grouped in a link graph (site map). When running a link checking routine, one usually finds not only links that point to non-existing pages, but also pages which are not interlinked with others or so-called orphan pages. An orphan page can be reached via a link, but doesn't have a link back to the hypertext structure.

3. Browser Testing : A large number of different Web browsers can be used as the client for Web applications. Depending on the manufacturer (e.g., Microsoft, Mozilla, Netscape, Opera), or the version (e.g., Internet Explorer 5.0, 5.01, 5.5, 6.0), or the operating system (e.g., Internet Explorer for Windows XP/2000, Windows 98/ME/NT, or Macintosh), or the hardware equipment (e.g., screen resolution and color depth), or the configuration (e.g., activation of cookies, script languages, stylesheets), each Web browser shows a different behavior.

Browser testing tries to discover errors in Web applications caused by incompatibilities between different Web browsers. To this end, one normally defines a Web application's core functions, designs suitable test cases, and runs the tests on different target systems with different browser versions.

4. Usability Testing : Usability testing evaluates the ease-of-use issues of different Web designs, overall layout, and navigations of a Web application by a set of representative users. The focus is on the appearance and usability. A formal usability test is usually conducted in a laboratory setting, using workrooms fitted with one-way glass, video cameras, and a recording station. Both quantitative and qualitative data are gathered.

Usability testing involves following main steps:

- Identify the website's purpose.
- Identify the indented users.
- Define tests and conduct the usability testing
- Analyze the acquired information

5. Load Testing : A load test verifies whether or not the system meets the required response times and the required throughput. To this end, we first determine load profiles (what access types, how many visits per day, at what peak times, how many visits per session, how many transactions per session, etc.). Next, we determine the target values for response times and throughput. Subsequently, we run the tests, generating the workload with the transaction mix defined in the load profile, and measure the response times and the throughput. The results are evaluated, and potential bottlenecks are identified.

6. Stress Testing : Stress testing consists of subjecting the system to varying and maximum loads to evaluate the resulting performance. We use authorization tests tools to simulate loads on a website and execute the tests continuously for several hours or days,

7. Form testing : Website that uses forms need tests to ensure that each field works properly and that the forms post all data as intended by the designer.

8. Page Content Testing : Each web page must be tested for correct content from the user perspective. These tests fall into two categories: ensuring that each components function correctly and ensuring that the content of each is correct.

9. Configuration and Compatibility Testing : A key challenge for the web application is ensuring that the user sees a web page as a designer intended. The user can select different browser software and browser options, use different network software, and on-line service, and run other concurrent application. We execute the application under every browser/platform combination to ensure the website work properly under various environments.

10. Reliability and Availability Testing : A key requirement for a website is that it would be available whenever the users request it, after 24 hours, a day. The number of user accessing website simultaneously may also affect the site availability.

11. Performance Testing : Performance Testing, which evaluates system performance under normal and heavy usage is crucial to the success of any web application. A system that takes for long to respond may frustrate the user who can then quickly move to a competitor's site. Given enough time, every page request will eventually be delivered. Performance testing seeks to ensure that the website's server responds to browser request with in defined parameters.

12. Security Testing : Security is a primary concern when communication and conducting business specially sensitive and business critical transaction over the internet. Then user wants assurance that personal and financial information is secure. Finding the vulnerabilities in an application that would grant an unauthorized user access to the system is important.

1.8 WEB CHARACTERISTICS

The essential feature that led to the explosive growth of the web - decentralized content publishing with essentially no central control of authorship - turned out to be the biggest challenge for web search engines in their quest to index and retrieve this content. Web page authors created content in dozens of

(natural) languages and thousands of dialects, thus demanding many different forms of stemming and other linguistic operations. Because publishing was now open to tens of millions, web pages exhibited heterogeneity at a daunting scale, in many crucial aspects. First, content-creation was no longer the privy of editorially-trained writers; while this represented a tremendous democratization of content creation, it also resulted in a tremendous variation in grammar and style (and in many cases, no recognizable grammar or style). Indeed, web publishing in a sense unleashed the best and worst of desktop publishing on a planetary scale, so that pages quickly became riddled with wild variations in colors, fonts and structure. Some web pages, including the professionally created home pages of some large corporations, consisted entirely of images (which, when clicked, led to richer textual content) - and therefore, no indexable text.

What about the substance of the text in web pages? The democratization of content creation on the web meant a new level of granularity in *opinion* on virtually any subject. This meant that the web contained truth, lies, contradictions and suppositions on a grand scale. This gives rise to the question: which web pages does one trust? In a simplistic approach, one might argue that some publishers are trustworthy and others not - begging the question of how a search engine is to assign such a measure of trust to each website or web page. More subtly, there may be no universal, user-independent notion of trust; a web page whose contents are trustworthy to one user may not be so to another. In traditional (non-web) publishing this is not an issue: users self-select sources they find trustworthy. Thus one reader may find the reporting of *The New York Times* to be reliable, while another may prefer *The Wall Street Journal*. But when a search engine is the only viable means for a user to become aware of (let alone select) most content, this challenge becomes significant.

While the question “how big is the Web?” has no easy answer, the question “how many web pages are in a search engine’s index” is more precise, although, even this question has issues. By the end of 1995, Altavista reported that it had crawled and indexed approximately 30 million *static web pages* . Static web pages are those whose content does not vary from one request for that page to the next. For this purpose, a professor who manually updates his home page every week is considered to have a static web page, but an airport’s flight status page is considered to be dynamic. Dynamic pages are typically mechanically generated by an application server in response to a query to a database, as show in Figure 1.3. One sign of such a page is that the URL has the character “?” in it. Since the number of static web pages was believed to be doubling every few months in 1995, early web search engines such as Altavista had to constantly add hardware and bandwidth for crawling and indexing web pages.

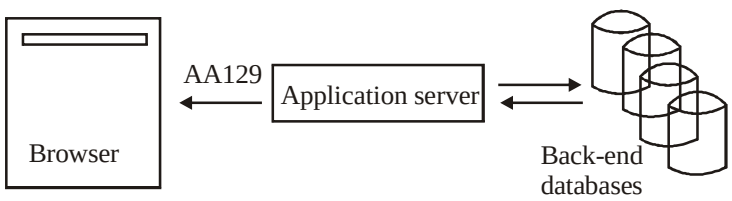


Fig. 1.3

A dynamically generated web page.The browser sends a request for flight information on flight AA129 to the web application, that fetches the information from back-end databases then creates a dynamic web page that it returns to the browser.

1.9 EVOLUTION AND NEED FOR WEB ENGINEERING

Modern Web applications are full-fledged, complex software systems. Therefore, the development of Web applications requires a methodologically sound engineering approach. Based on Software Engineering, Web Engineering comprises the use of systematic and quantifiable approaches in order to accomplish the specification, implementation, operation, and maintenance of high-quality Web applications. Web applications can have document centric, interactive, transactional, or ubiquitous characteristics, or even features of the semantic Web. The particular requirements of Web Engineering result from the special characteristics of Web applications in the areas of the software product itself, its development, and its use. Evolution is a characteristic that encompasses these three areas.

In the absence of a disciplined approach to Web-based system development, we will find sooner or later that Web-based applications are not delivering desired performance and quality, and that development process becomes increasingly complex and difficult to manage and refine and also expensive and grossly behind schedule.

"Web Engineering", is an emerging new discipline, advocates a process and a systematic approach to development of high quality Internet- and Web-based systems.

We provide a broad and objective definition of Web engineering as follows.

> *Web engineering is the establishment and use of sound scientific, engineering and management principles and disciplined and systematic approaches to the successful development, deployment and maintenance of high quality Web-based systems and applications.*

Web engineering principles and approaches can bring the potential chaos in Web-based system development under control, minimize risks, and enhance maintainability and quality.

Highlights:

- Provides insights into current concepts, methods, techniques, tools, and experiences for an engineering approach to Web application development.
- Complements existing technology-oriented books with an engineering approach.
- Identifies potential risks in Web application development.
- Identifies similarities and differences between the development of traditional, not Web-based applications and the development of Web applications.
- Analyses concepts, methods, techniques, and tools of traditional software engineering to see how suited they are for Web application development.
- Defines an important new vocational discipline, popular with students.

- Offers teachers a viable alternative to teaching software engineering.
- Reports on future developments in Web Engineering.

1.10 WEB ENGINEERING VERSUS SOFTWARE ENGINEERING

Contrary to the perception of some software developers and software engineering professionals, Web engineering isn't a clone of software engineering although both involve programming and software development. While Web engineering adopts and encompasses many software engineering principles, it incorporates many new approaches, methodologies, tools, techniques, and guidelines to meet the unique requirements of Web-based systems. Developing Web-based systems is significantly different from traditional software development and poses many additional challenges. There are subtle differences in the nature and life cycle of Web-based and software systems and the way in which they're developed and maintained. Web development is a mixture between print publishing and software development, between marketing and computing, between internal communications and external relations, and between art and technology.

Though Web engineering involves some programming and software development, and adopts some of the principles of the software engineering, Web-based system development is different from software development, and also Web engineering is different from software engineering.

1. Most Web-based systems, at least as of now, are document-oriented containing static or dynamic Web pages.
2. Web-based systems will continue to be focused on look and feel, favoring visual creativity and incorporation of multimedia (in varying degrees) in presentation and interface. More emphasis will be placed on visual creativity and presentation as regards to the front-end interface with which a user interacts.
3. Most Web-based systems will continue to be content-driven–often Web-based systems development include development of the content presented.
4. Multiplicity of user profiles–Most Web-based systems need to cater to users with diverse skills and capability, complicating human-computer interaction, user interface and information presentation.
5. The nature and characteristics of the medium of Web is not well understood as the software medium.
6. The Web exemplifies a greater bond between art and science than generally encountered in software development.
7. Most Web-based systems need to be developed within a short time, making it difficult to apply the same level of formal planning and testing as used in software development.
8. Also Web is different from software as related to the delivery medium.
9. Further, the type of individuals who build/develop Web-based systems are vastly varied in their background, skills, knowledge and system understanding, and as well as their perception of Web and quality Web-based system.

1.11 WEB ENGINEERING AND WEB GARDENING

Many Web-based systems call for continual update or refinement, and hence Web-based system development may be considered as "continuous, with fine grained evolution, without specific releases as with software." In this respect, Web-based system development is like gardening like a garden, Web-based system will continue to evolve, change and grow. However, a good initial infrastructure is required to allow the growth to occur in a controlled, but flexible and consistent manner, and to foster creativity, refinement and change.

The garden analogy to Web-based system development and the nature of Web as a flexible medium may make us think, or wonder, for a moment whether Web engineering approaches are appropriate for Web-based system development. We believe that they are appropriate, as they are adapted to Web environment and provide flexibility to work within a framework and allow creative development.

They are not as 'rigid' as perceived by some based on their perception of some of the 'traditional engineering' approaches, and allows creativity and personalization to blossom within a framework/limited boundaries. In fact, all that Web engineering advocates is "use of sound scientific, engineering and management principles and disciplined and systematic approaches to the successful development, deployment and maintenance of high quality Web-based systems and applications." It is appropriate provided we make sure that the approaches are appropriate to the Web environment. Both the Web engineering and Web gardening metaphors are valid in Web environment, and perhaps we may need to follow what is appropriate from both the approaches.

1.12 WEB ENGINEERING SKILLS

Given the state of Web engineering, the most influential factor for the success of Web engineering projects is the skill level required from Web engineers to master the development process. Thus, an important step in teaching Web engineering consists of identifying the skills that one expects Web engineers to possess. Groups of skills are required for any software engineer. Likewise, Web engineers are expected to possess three groups of skills:

1. The first group, prerequisite skills, is required for any student entering the field of Web engineering. The skills must be acquired at an early stage, because the learning of Web engineering depends on.
2. The second group is concerned with specific Web engineering skills. It provides the ability to perform key tasks of the Web engineering development process.
3. The third group, generic skills, is concerned with writing, reading, communication, dialogue, teamwork, and project planning. These skills are essential for Web engineers in a work situation.

PREREQUISITE SKILLS

1. Object-oriented modeling and programming with UML and Java, or similar languages.
2. Database development with the JDBC, MySQL and Java Servlets, or similar languages.

3. Web programming with HTML, JavaScript, CGI Script, and PHP, or similar languages.
4. Deployment of specialized authoring tools and Web editors, such as FrontPage and Macromedia Dreamweaver for the design and implementation of Web-based applications.

SPECIFIC SKILLS

1. Understanding Web engineering as a multidisciplinary field incorporating technical, engineering, social, political, marketing, legal, ethical, cultural, esthetical, and pedagogical issues.
2. Understanding the Web engineering development philosophy.
3. Understanding the context and system scope of Web engineering applications.
4. Analysis modeling: understanding the problem requirements; specifying users' requirements using use cases and scenarios; specifying data requirements using classes and other analysis modeling techniques; and general quality attributes for Web-based applications.
5. Design modeling: architecture design; component, logical, and deployment design; cohesion and coupling issues; elaboration of requirements and design criteria.
6. Web design: site design using linear, grid, hierarchical, and network Web structures; page design; interface design; navigation design, visual design; design of typography, editorial style, screen, colors, graphics, and multimedia.
7. Prototyping, incremental, evolutionary, and iterative development; program coding, unit and integration testing, evaluating and debugging of the evolving code solution, and consistency checking.
8. Usability testing and engineering; human-computer interaction; usability criteria in terms of user satisfaction, ease-of-use; ease-of-learn, and consistency.
9. Reflecting, evaluating, explaining, and justifying Web engineering solutions.

GENERIC SKILLS:

1. Project planning and management.
2. Reuse of design principles, frameworks, architectures, and toolkits (class libraries); modifying and reusing existing analysis, design and program code solutions and patterns; comparing, contrasting, recognizing similarities and differences between new problems and previous solutions.
3. Writing and reading skills: writing and formatting technical documentation and reports, reading texts and documents.
4. Dialogue and communication with stakeholders. Working in teams with developers, system designers, programmers, end-users, and clients.

SUMMARY

1. The operation of the web relies mainly on hypertext as its means of interacting with users. Hypertext is basically the same as regular text – it can be stored, read, search or edited – with an important exception: hypertext contains connections within the text to other documents.
2. Hypermedia is hypertext with a difference – hypermedia documents contain links not only to other pieces of text, but also to other forms of media – sounds, images, and movies. Images themselves can be selected to link to sounds or documents.
3. Hypermedia simply combines hypertext and multimedia.
4. The internet is the catch-all world used to describe the massive worldwide network of computers. The word "internet" literally means "network of network".
5. The World Wide Web is mostly used on the internet; they do not mean the same thing. The web refers to a body of information – an abstract space of knowledge, while the internet refers to the physical side of the global network, a giant mass of cables and computers.
6. The World Wide Web is mostly used part of the internet by far. Once you spend time on the web you'll feel that there is no limit to what you can discover.
7. In June 1993, Mathew Gray at MIT ran a small program which automatically travels links within the Web network to try to determine just how many sites there are that offer information over the World Wide Web.
8. The World Wide Web exists virtually – there is no standard way of viewing it or navigating around it. However, many software interfaces to the Web have similar functions and generally work the same way no matter what computer or type of display is used.
9. The HTTP protocol used by web browser to talk to web server, the "file extension" of the URL is not used to determine the type of information that the server will return.
10. HTML, an acronym for Hyper Text Markup Language, is the predominant markup language for web pages. It provides a means to describe the structure of the text-based information in a document–by denoting certain text as links, headings, paragraphs, lists, etc.

Short Question Answers

Q.1. How do we analysis Web Traffic.

Ans. Once your Web site is up and running, you'll probably have an ongoing interest in knowing how well it is attracting visitors. The Web server program creates an entry in a log file every time the server responds to a request for an HTML document, graphic, form, or other service. This means that every user is tracked during every visit to your site. A number of traffic analyses Programs on the market can crunch these huge files to produce tables and graphics that summarize a Web server's act.

Q.2. What are search Engine? How do they work? Give three example of search engine?

Ans. Search Engine : A search engine is a database application that retrieves

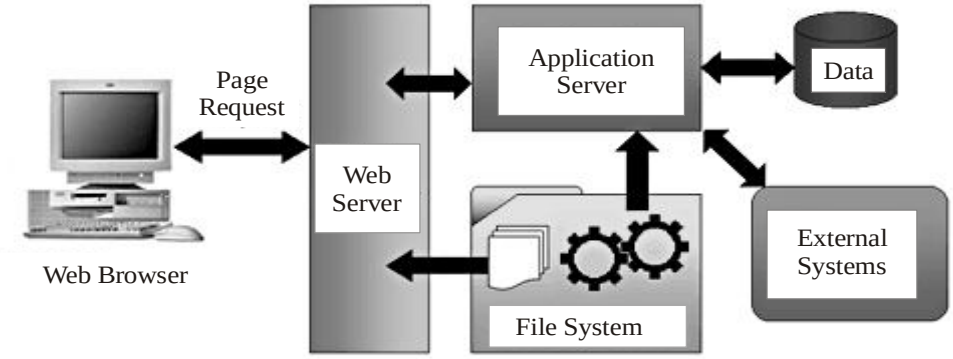
information, based on words or a phrase that you enter. A Web search engine employs a search agent (also called a spider) that goes out and looks for information on Web pages. This information is indexed and stored in a huge database. When you conduct a search, the search engine looks through its database to find entries that match the information you entered. Then, the search engine presents to you a list of the Web pages that it determines are most relevant to your search criteria.

Some example of search engines:

1. AltaVista
2. Google
3. Yahoo.

Q.3. Explain the structure of the web in detail.

Ans. The structure of the Web is a lattice of HTML-formatted pages, distributed from computers known as Web servers to client computers, and viewed using tools called Web browsers. At the most basic level, there is no difference between a server that has thousands of pages and a server that just has one or two. It is by linking these pages together in some form that a set of pages becomes a Web site, and by adding some additional logic, a Web site becomes a Web application.



- **Web Page :** The basic unit of a Web interaction is the Web page itself. A Web page is a text file that is marked up using HTML. It is sent to a browser from a Web server based on a request from that browser. The browser parses the information in the HTML file, and the resulting user interface is displayed within the browser itself. The role of the Web server is to listen for a request from the client, parse the request to determine the page that the client requested, retrieve that file from the server's storage area, then transmit that file to the client. At this point, the server forgets everything about sending that file to the client, except for maybe placing an entry into a log file. It is this "connection-less" nature that gives a Web server its scalability, but in turn makes it a challenge to create meaningful applications without some additional support.

- **Web Site :** A Web site consists of a set of related Web pages grouped together by some means. Generally, a Web site is all of the pages that exist on a server, or within a folder on that server. For example, all of the pages that are on the http://www.wrox.com server are considered part of that Web site. The correlation between

the pages on a site is maintained by the links within each page on the site. The links on each page in the site will take users to other pages within the site. In this way, the pages that make up the site internally maintain the hierarchy of the site. These pages are still subject to the restriction of the Web server architecture, in that the server does not maintain information about the series of requests from a particular client. So while this related set of Web pages that make up a Web site are beginning to look more like an application, there are still some missing components.

- **Web Applications :** Windows DNA applications are applications in the traditional sense, in that they provide a service to the user of the application. They are different in the way that they are created as well as in the components that make them up. A traditional application requires a special set of files during development, but distributes different outputs. For example, a Visual Basic application has a .vbp project file, multiple .frm, .cls, and .bas files, as well as a set of OCX components that make up the application project. Prior to the application being distributed, these files are compiled into a set of executable files for distribution and execution. The resulting executable does not require the presence of the source code files that were used to develop it.

Script-based Web applications, on the other hand, are composed of the same set of files used during development and after deployment. There is no compiled executable file produced that becomes the Web application. For example, the .htm, .asp, and .dll files in your Web project are the same files you deliver to your production Web server. The source code, or script, in these Web files is executed on the client or server only when a browser requests the Web page. Creating these applications builds upon the architecture of the World Wide Web, but there is some added complexity and functionality in order to have these files function as an application.

- **Web Application Design :** In building a Web application, there are a number of new aspects of application design and development that the developer must take into consideration. While the flexibility of Windows DNA allows for a wide variety of clients accessing a wide variety of services, we will be focusing on a Web application. In a Web application, we will look at using a browser as the primary user interface. The information in our Web application will flow from server to client using the HTTP protocol. Our Application server will be Microsoft's Internet Information Server functioning as both a Web and Application server. Finally, our business and data access logic will be encapsulated within COM+ components and serve as the plumbing that links our application together.

- **The Browser as the User Interface :** For a Web application, the user interface is presented within a Web browser. This means that the client presentation can either be Browser-Enhanced or Browser-Reliant. The type that you choose for your application should be based on the installed browser base of your target audience. If you can guarantee that most of your users will be using a particular browser level, then you should consider leveraging the features of that browser level. If there are a wide variety of browsers in use, then you may only be able to support a Browser-Reliant client presentation type. One of the advantages of using Active Server Pages to deliver the application is that it has the capability to determine with what browser the current user is accessing the application. This was covered

in Chapter 6 with the Browser Capabilities component. By knowing the browser type of the current user, you can dynamically change your client presentation to support the enhanced characteristics of that browser.

- **HTTP as the Transport:** The communication layer between the client and the server is critical to the design and implementation of the application. The HyperText Transport Protocol (HTTP) defines how the request made by the client is received and handled by the server, and then how the information is sent back to the client. Depending on the type of client presentation being supported, there can be different types of information that flow using this protocol. For the basic support of a Browser-Reliant client, the information that flows over HTTP is limited to the HTML that makes up the page, graphical images to enrich the interface presentation, information-bearing cookies, and possibly some client-side scripting to provide interactivity on the client. With an Enhanced client, special features such as Remote Data Services or COM over HTTP may also be communicated over the transport protocol. But support of these is reliant on the capabilities of the browser at the client. In either case, the HTTP protocol does not understand the concept of a persistent connection between client and server.

- **IIS and ASP as the Application Server :** The combination of Microsoft's Internet Information Server as the Web server and Active Server Pages as the Application server provides the application component of our Web application. For Web pages and Web sites that serve up static Web pages for display, IIS can function by itself to provide the information. But when a Web application demands a dynamic display of information, we need to link the Web serving capabilities of IIS with the dynamic page generation and object integration capabilities of ASP to deliver a more robust and dynamic Web application. The scripting capabilities of ASP allow us to support business logic inside of our scripts, link with business logic components, directly access databases via ADO, and make use of data components to retrieve information.

Q.4. What is firewall? Explain its importance.

Ans. The hardware, software, and procedures that provide access control make up a firewall. A firewall can limit Internet access to e-mail only, so that no other types of information can pass between the intranet and the Internet. Firewall systems fall into two categories,

- network-level
- application-level.

Network-Level Firewalls: These firewalls examine only the headers of each packet of information passing to or from the Internet. The firewall accepts or rejects packets based on the packet's sender, receiver, and port.

Application-Level Firewalls: These firewalls handle packets for each Internet service separately, usually by running a program called a proxy server, which accepts e-mail, Web, chat, newsgroup, and other packets from computers on the intranet, strips off the information that identifies the source of the packet, and passes it along to the Internet. When the replies return, the proxy server passes the replies

back to the computer that sent the original message. A proxy server can also log all the packets that pass by, so that you have a record of who has access to your intranet from the Internet, and vice versa.

Q.5. Write short note on:

(i) E-mail (ii) E-Commerce (iii) Plug ins and Active X.

Ans. E-mail:- email is also called Electronic mail. It is a method of exchanging digital messages across the Internet or other computer networks. Today's email systems are based on a store-and-forward model. Email servers accept, forward, deliver and store messages. Users no longer need be online simultaneously and need only connect to an email server, for as long as it takes to send or receive messages. An email message consists of two components, the message header, and the message body. The message header contains control information, including, generally, a sender's email address and one or more recipient addresses. Usually additional information is added, such as a subject header field. Email is carried by the Simple Mail Transfer Protocol (SMTP), first published as Internet standard

E-Commerce : Electronic commerce, commonly known as e-commerce, consists of the buying and selling of products or services over electronic systems such as the Internet and other computer networks. The amount of trade conducted electronically has grown with widespread Internet usage. A large percentage of electronic commerce is conducted entirely electronically for virtual items such as access to premium content on a website, but most electronic commerce involves the transportation of physical items in some way. Online retailers are sometimes known as e-tailors and online retail is sometimes known as e-tail.

Plug-ins and Active X: Sometimes, a player can't handle the information on a Web page. Players work only if information is stored in a separate file and you want it to appear in a separate window, not in your browser window. Hence two other types of programs: plug-ins and ActiveX controls are used to handle audio, video, and other information right in your browser A plug-in is a program that can "plug in" to your browser, to give the browser a new capability. Most plug-ins works with Internet Explorer, NN and some work with Opera too. Plug-ins works with browser seamlessly, so that after you install them, you can forget that they are not part of your browser. Netscape originally invented the idea of plug-ins, but Microsoft makes sure that Internet Explorer can use most of them, too.

Q.6 What is cookies? Explain its importance.

Ans. A cookie is a small file that a Web server can store on your machine. Its purpose is to allow a Web server to personalize a Web page, depending on whether you have been to that Web site before, and what you may have told it during previous sessions. When you return to that Web site in the future the Web server can read its cookie, recall this information, and structure its Web pages accordingly. However, cookies do make it easier for advertising companies to gather information about your browsing habits.

Q.7 Describe the process of Web Publishing phases of website development.

Ans. Web publishing, or "online publishing," is the process of publishing content on the Internet. It includes creating and uploading websites, updating web pages, and posting blogs online. The published content may include text, images, videos, and other types of media. Publishing Process: The three essential steps to publish your website onto the web are:

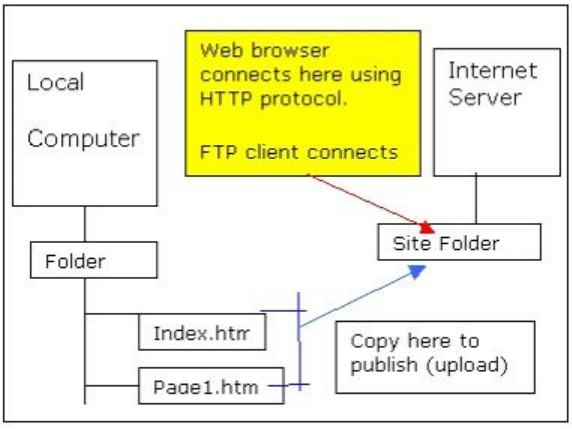
1. **Creating your website:** First you have to create a website. Depending on your level on knowledge in web development you have the following options. Either you can create the site yourself through manual programming in an editor program like DreamWeaver, EditPlus or Notepad.

2. **Get a hosting plan:** Next step is to find somewhere online to put your website files and assign an address to that somewhere. Any given hosting plan include storage space on a web server and a domain name registration.

3. **Upload your files/Publish your website:** Upload your files to the root directory of your purchased web hosting server space. You can do this either by using your hosting account's inhouse file manager, but those are often slow and manage large file volume poorly. Our recommendation is to get your hands on a FTP client.

Q.8. What do you understand by uploading of a webpage? How can it be done?

Ans. Uploading web pages to the Internet makes them accessible to Internet users. You normally create the Website in a folder located on your local computer. Uploading or Publishing a web page/pages simply copies the page from your local computer to the Internet. Normally you upload web pages using the FTP protocol, however most of the HTML editors have their own in built publishing mechanism. Uploading Web Pages - When we upload (publish) a Web page we simply copy the file from a folder on our local computer to a folder on a computer on the Internet. Our Website is simply a folder containing files. Normally when you first create a website, all of the files in the local folder are uploaded to the Internet server. Subsequent updates are done by uploading individual files/pages to either replace existing ones or to add to the site content.



FTP (File transfer Protocol) is the most common method used to upload web pages to the Internet server. The process is simply a file copy from your PC to the web server.

The general procedure is as follows:

1. Connect to server

2. Upload files to server.
3. Login by entering username and password.
4. Verify that files have been transferred.
5. Logout.

Q.9 Write all the steps of hosting a website using FTP.

Ans. To transfer the file across the Internet, you have several choices:

- Transfer the file using one of the Internet services: e-mail or the Web.
- Use software specially designed to transfer "any file," software that uses a special protocol called File Transfer Protocol (FTP).

Connecting to the FTP Server : To connect to an FTP server, you give the host name of the server to your FTP client software and then you log in. You can log in to an FTP server in one of two ways:

If you have an account on the FTP server, you log in with a user name and password. You can access all the files that your user name gives you permission to use. If you don't have an account on the FTP server, you can log on anonymously. Once you have logged in to an FTP server, the server may display welcoming and instructional text about using the server.

FTP servers transmit messages to let you know what's going on. Once you are Connected to an FTP server, you select a particular folder, called the current working directory, from which you will download, or to which you will upload, files. If you have permission to do so, you may be able to create additional folders, rename folders, or delete them. To upload a single file, use the put command. To upload a group of files, use the mput command.

Q.10. What is Markup ?

Ans. The concept of markup originates from the publishing industry and generally means typographic instructions for document formatting. These instructions are specified inside a document in the form of additional characters. For example, we could write *Hello* to output Hello or /Hello/ to output Hello. Semantic markup lets us write comments within text without showing them in the document. ISO defines the following markup classes:

1. Markup : This is text inserted in a document to add information as to how characters and contents should be represented in the document.

2. Descriptive markup: This is markup that describes the structure and other attributes of a document, regardless of how this document is processed for representation (e.g., comments).

3. Processing instructions : This is markup consisting of system-specific data; it controls the way a document is processed.

Q.11. Explain the following in brief ?

(i) Telnet (ii) FTP (iii) XML (iv) ASP (v) Applets (vi) HTTP

Ans. Telnet : TELNET (TErminaL NETwork) is a network protocol used on the Internet or local area networks to provide a bi-directional interactive text-oriented communications facility via a virtual terminal connection. Historically, telnet provided access to a command-line interface on a remote host. Most network equipment and operating systems with a TCP/IP stack support a Telnet service for remote configuration. The term telnet may also refer to the software that implements the client part of the protocol. Telnet client applications are available for virtually all computer platforms. Telnet means to establish a connection with the Telnet protocol, either with command line client or with a programmatic interface.

FTP : File Transfer Protocol (FTP) is a standard network protocol used to copy a file from one host to another over a TCP/IP-based network, such as the Internet. FTP is built on a client–server architecture and utilizes separate control and data connections between the client and server applications which solves the problem of different end host configurations. FTP is used with user-based password authentication or with anonymous user access.

XML : The Extensible Markup Language (XML) is a W3C recommendation for creating special-purpose markup languages. It is a simplified subset of SGML, capable of describing many different kinds of data. Its primary purpose is to facilitate the sharing of structured text and information across the Internet. XML (Extensible Markup Language) is a set of rules for encoding documents electronically. It is defined in the XML 1.0 Specification produced by the W3C, and several other related specifications. XML's design goals emphasize simplicity, generality, and usability over the Internet. It is a textual data format, with strong support via Unicode for the languages of the world. Although XML's design focuses on documents, it is widely used for the representation of arbitrary data structures in web services. There are many programming interfaces that software developers may use to access XML data, and several schema systems designed to aid in the definition of XML-based languages.

ASP : An Active Server Page (ASP) is an HTML page that includes one or more scripts (small embedded programs) that are processed on a Microsoft Web server before the page is sent to the user. An ASP is somewhat similar to a server-side include or a common gateway interface (CGI) application in that all involve programs that run on the server. Typically, the script in the Web page at the server uses input received as the result of the user's request for the page to access data from a database and then builds or customizes the page before sending it to the requestor.

Applets : Some applets are able to function as any other normal software application, but are small in size and perform only a small set of tasks. Applets are not full-featured application programs. In some cases, an applet does not run independently. Such applets must run in a container, which is provided by a host program, through a plug in, or a variety of other applications including mobile devices that support the applet programming model. Java programmers usually include applets

by using the statement import java.applet.Applet

HTTP : The Hypertext Transfer Protocol (HTTP) is an application-level protocol necessary for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol which can be used for many tasks, such as name servers and distributed object management systems, through extension of its request methods. A feature of HTTP is the typing of data representation, allowing systems to be built independently of the data being transferred.

MULTIPLE CHOICE QUESTIONS

1. In which year the internet is founded:
   a. 1969 b. 1990
   c. 1972 d. 1995
2. In which year the concept of web are begin:
   a. 1968 b. 1980
   c. 1989 d. 1890
3. Website is a collection of :
   a. Web server b. Web pages
   c. Web browser d. Www
4. A homepage is also called:
   a. Web browser b. Website
   c. Web pages d. Index pages
5. Web publishing means:
   a. Collecting the web pages b. Publishing the web pages
   c. Storing the web pages d. None of these
6. Which of these searches new terms on internet and stores the searched terms in a database?
   a. Crawler b. Indexer
   c. Search algorithm d. None of these
7. _____ facilitates the access of hypertext document :
   a. TCP/IP b. FTP
   c. NNTP d. HTTP
8. DNS stand for :
   a. Domain Name Source b. Domain Name System
   c. Description Name System d. None of these

9. The unique address of each web page is called as:

a. URL b. FTP

c. HTML d. FIIT

10. URL is stand for:

a. Uniform research location b. Uniform resource locator

c. Universal research locator d. None of these

11. IP address is written as a string of _____ numbers separated by dots('.')

a. Four b. Three

c. Two d. One

12. The relationship b/w the ip addresses and the domain names is maintained by an internet service called the:

a. IP b. DNS

c. URL d. None

13. Scripting languages can be used to provide:

a. Simple features to the web page

b. Main features to theweb page

c. Dynamic features to the web page d. None of these

KEY TO OBJECTIVE QUESTIONS

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

IMPORTANT QUESTIONS

Q.1 Define Web Engineering. Draw a diagram to categorize web applications.

Q.2 Explain evolution and need of Web Engineering.

Q.3 Explain what are different Web Engineering Activities and Web Engineering Skills?

Q.4 Explain Web-based System Development .

Q.5. What are different Web Testing strategies ?

Q.6 What are the characteristics of Web Applications?

Q.7 Differential Software Engineering with Web Engineering?

Q.8 What are main organizational challenges to develop a web site.

Q.9 Write the steps to show how Search Engine works.

Q.10 Explain Web Engineering Discipline in details.

Q.11 Explain the structure of the web in detail.

Q.12 What do you understand by uploading of a webpage? How can it be done?

Q.13 Describe the process of Web Publishing phases of website development.

****

WORLD WIDE WEB

IN THIS CHAPTER, YOU WILL LEARN

- Introduction
- The World Wide Web (WWW)
- Web Browsers
- Web server
- Hypertext Transfer Protocol
- Domain Name System (DNS)
- URLs
- TCP/IP (Transmission Control Protocol/Internet Protocol)
- IP address (Internet Protocol Address)
- File Transfer Protocol (FTP)
- Searching Techniques
- Search Engine
- WAP–Introductio
- Internet Services
- Steps to Create Your Web Site
- Enhancing Your Website
- Good Web Design
- Home Page
- Web Designing
- Dynamic HTML
- Web Publishing Process
- Phases of Website Development
- Webcasting

2.1 INTRODUCTION

This is a virtual world provided by networks of computers with multi-users. It is an international network of networks of computers linking different types of users: Academic, Industries, Government, Health Institutions, military, individuals, etc, for the purpose of sharing information.

As a communication network among computers, the internet allow you to locate and retrieve information on other computers linked to the internet as well as send messages electronically to and from other people elsewhere on the internet. Whenever Internet software application is used, the client software will either be on your personal computer, the computer you log onto for access to the internet (your host), or yet another computer to which you connect in other to use client software you may not have on your computer. As you navigate through the Internet, you will find yourself logged onto different host computers, sometimes gaining access to different client programs and also accessing different servers; it can be complicated. Fortunately, the purpose of advanced Internet software is to hide these complexities from users so as to achieve success.

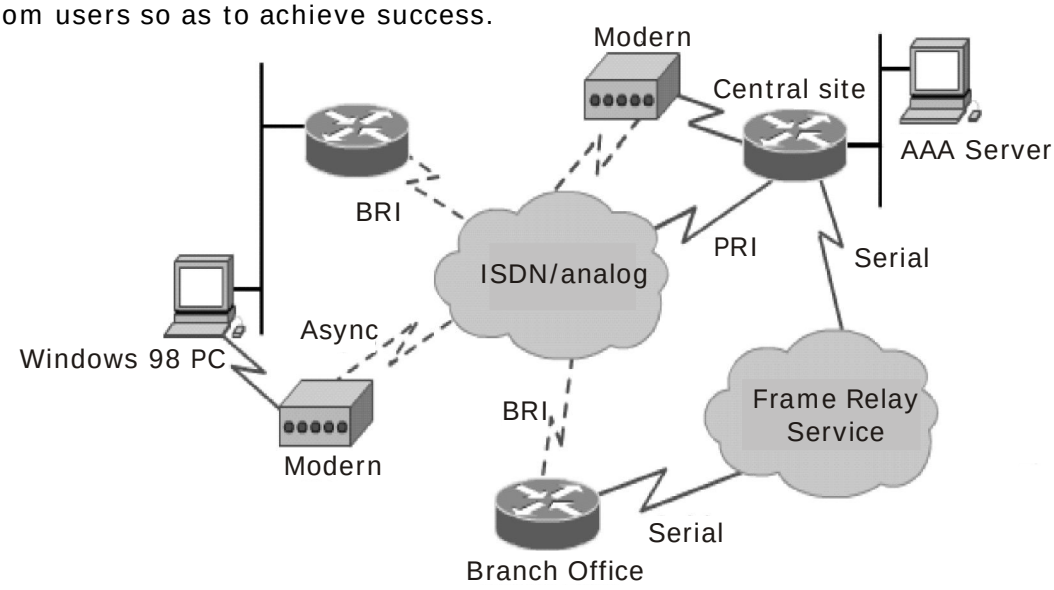


Fig. 2.1

The internet has six application protocols: Electronic mail (E-mail), World Wide Web (Hypertext Transfer Protocol or HTTP), Gopher, Telnet, File Transfer Protocol (FTP), Each of these application protocol has special client software, and many web browsers, such as Netscape from communication and internet Explorer from Microsoft, which are capable of reading and displaying data from all the applications.

Benefits of Internet Application :

- Downloading of information
- Advertisement
- Accessing newspapers, magazines and academic journals
- On-line banking
- Accessing international media (CNN, BBC, VOA)

2.2 THE WORLD WIDE WEB (WWW)

World Wide Web (often termed simply as "Web") is the most exciting new tool for the Internet. It is based on the technology called hypermedia. With hypermedia, information in one document can be linked to another; related information can consist of not only text but graphics, audio and video information as well. WWW is an ambitious, exciting and powerful attempt to link connected information wherever it may be located on the Internet, allowing the user to easily access and retrieve related files. The terms Internet and World Wide Web are often used in every-day speech without much distinction. However, the Internet and the World Wide are not one and the same. The Internet is a global data communications system. It is a hardware and software infrastructure that provides connectivity between computers. In contrast, the Web is one of the services communicated via the Internet. It is a collection of interconnected documents and other resources, linked by hyperlinks and URLs. These hyperlinks and URLs allow the web servers and other machines that store originals, and cached copies, of these resources to deliver them as required using HTTP (Hypertext Transfer Protocol). HTTP is only one of the communication protocols used on the Internet.

2.2.1 History

The World Wide Web (WWW or the Web) was originally created during the Cold War as a way to link sections of the country together during an emergency, the actual term "Internet" wasn't used until the early 1970s. At that time, academic research institutions developed the Internet to create better communication and to share resources. Later, universities and research facilities throughout the world began using the Internet. In the early 1990s, Tim Berners-Lee created a set of technologies that allowed information on the Internet to be linked together through the use of links, or connections, in documents. The language component of these technologies is Hypertext Markup Language (HTML).

The Web was mostly text based until Marc Andreessen created the first graphical Web browser in 1993, called Mosaic. This paved the way for video, sound, and photos on the Web.

As a large group of interconnected computers all over the world, the Internet comprises not only the Web, but also things like newsgroups (online bulletin boards) and e-mail. Many people think of the Web as the graphical or illustrated part of the Internet.

2.2.2 World Wide Web Features

- The amount of information available on the Internet has become so large that it is difficult to search for specific information. The World Wide Web (WWW) makes retrieval easy and quick.
- The WWW is a search tool that helps you find and retrieve information from a Web site using links to other sites and documents. The WWW was built on the technology called Hypertext. This technology increases accessibility to linked documents on the Internet and helps user to navigate between documents very easily.
- Hypertext is identified by underlined text and a different color usually. Some places will refer to these types of technique as Jump-Off Points. Hypertext can make links within the same document or to other documents.
- Each time you access a new document by choosing a link, there is a connection made with the web server that the document is on. Once the appropriate document is retrieved the connection is broken. There is no point in maintaining the link while you are viewing it. This is one reason why the WWW is so efficient.
- WWW lets you search, traverse, and use many types of information at numerous sites and in multiple forms. This interface is called a browser. Some people refer to a browser as a 'web browser' often these terms are used interchangeably.
- The WWW is intended to help people share information resources, and services with the widest possible community of users. Thus a user can access the WWW on Apple, UNIX, Macintosh, DOS, Windows, and other operating systems.
- Just like the Internet, the WWW has a protocol, which is known as HyperText Transfer Protocol (HTTP). HTTP acts as an interface between a Web Client Software, such Netscape Navigator.
- A major advantage of the WWW is that it also supports TCP/IP services, such as Gopher, FTP, and Archie in addition to HTTP.

2.2.3 How A Web Page Works

Viewing a Web page on the World Wide Web normally begins either by typing the URL of the page into a Web browser, or by following a hyperlink to that page or resource. The Web browser then initiates a series of communication messages, behind the scenes, in order to fetch and display it.

First, the server-name portion of the URL is resolved into an IP address using the global, distributed Internet database known as the domain name system, or DNS. This IP address is necessary to contact and send data packets to the Web server.

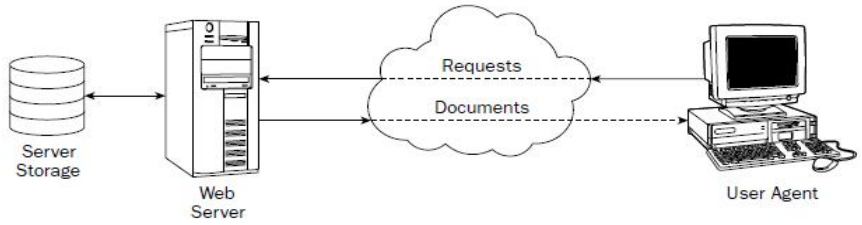


Fig. 2.2

The browser then requests the resource by sending an HTTP request to the Web server at that particular address. In the case of a typical Web page, the HTML text of the page is requested first and parsed immediately by the Web browser, which will then make additional requests for images and any other files that form a part of the page. Statistics measuring a website's popularity are usually based on the number of 'page views' or associated server 'hits', or file requests, which take place.

Having received the required files from the Web server, the browser then renders the page onto the screen as specified by its HTML, CSS, and other Web languages. Any images and other resources are incorporated to produce the on-screen Web page that the user sees.

Most Web pages will themselves contain hyperlinks to other related pages and perhaps to downloads, source documents, definitions and other Web resources. Such a collection of useful, related resources, interconnected via hypertext links, is what was dubbed a "web" of information. Making it available on the Internet created what Tim Berners-Lee first called the World Wide Web.

2.3 WEB BROWSERS

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

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

All browsers include bookmarks (Favorites) that store the addresses (URLs) of frequently used pages. Tabs are another useful feature that keep multiple Web pages open for quick access. The most popular Web browsers are Internet Explorer (IE), Firefox, Chrome, Safari and Opera. All browsers are free, and except for IE, which is Windows-only, they run on both Windows and Mac. Some browsers also run under Linux.

All browsers offer similar features, no matter which computer they run on. The way users interact with a Web page has more to do with the page than the browser. Web pages contain embedded programs that turn them into applications not much different than the software users install in their own computers.

How a Browser Retrieves a Web Page

The browser application retrieves or fetches code, usually written in HTML (HyperText Markup Language) and/or another language, from a web server, interprets this code, and renders (displays) it as a web page for you to view. In the majority of cases,

user interaction is needed to tell the browser what web site or specific web page he or she would like to view. One way this is done is via the browser's address bar. The web address or URL (Uniform Resource Locator), that you type into the browser's address bar tells the browser where to obtain a page or pages from.

2.4 WEB SERVER

Web Server is a computer that runs a Web site. Using the HTTP protocol, the Web server delivers Web pages to browsers as well as other data files to Web-based applications. The Web server includes the hardware, operating system, Web server software, TCP/IP protocols and site content (Web pages, images and other files). If the Web server is used internally and is not exposed to the public, it is an "intranet server".

2.4.1 HTTP Server

The term "Web server" often refers only to the HTTP server software in the machine, which provides the Web site functionality. HTTP is the protocol of the Web, and HTTP server software, such as Microsoft's IIS and the open source Apache server, accepts requests from the user's browser and responds by sending back HTML documents (Web pages) and files. It also executes scripts that reside in the server (CGI scripts, JSPs, ASPs, etc.), which perform functions such as database searching and credit card authorization.

Web servers are not only on the Web. HTTP server software is commonly built into hardware to provide a control panel for configuring the device from any Web browser. Most network devices such as routers, access points and print servers actually contain a mini Web site for this purpose.

2.4.2 Web Servers Working

Through an Internet connection, your browser initiates a connection for the page in www.google.com to the Web server that is storing the google files by first converting the domain name into an IP address (through a domain name service) and then locating the server that is storing the information for that IP address.

The Web server stores all of the files necessary to display google's pages on your computer typically all the individual pages that comprise the entirety of a Web site, any images/graphic files and any

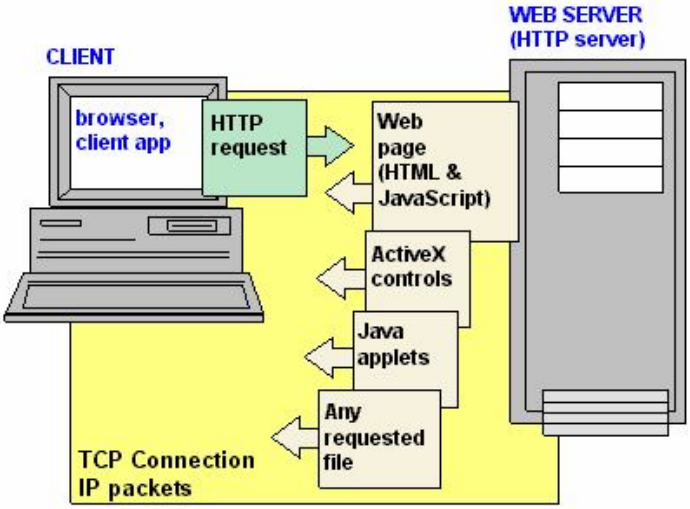


Fig. 2.3

scripts that make dynamic elements of the site function.

Once contact has been made, the browser requests the data from the Web server, and using HTTP, the server delivers the data back to your browser. The browser in turn converts, or formats, the computer languages that the files are made up of into what you see displayed in your browser. In the same way the server can send the files to many client computers at the same time, allowing multiple clients to view the same page simultaneously.

2.5 HYPERTEXT TRANSFER PROTOCOL

The **Hypertext Transfer Protocol (HTTP)** is an application protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web. This communications protocol used to connect to Web servers on the Internet or on a local network (intranet). Its primary function is to establish a connection with the server and send HTML pages back to the user's browser. It is also used to download files from the server either to the browser or to any other requesting application that uses HTTP.

Addresses of Web sites begin with an **http://** prefix; however, Web browsers typically default to the HTTP protocol. For example, typing **www.yahoo.com** is the same as typing **http://www.yahoo.com**. In fact, only **yahoo.com** has to be typed in. The browser adds the rest.

- **HTTP vs. HTTPS :** With HTTP, the Web page is transmitted without any encryption. However, HTTPS (HTTP Secure) is to access a secure Web server. When https:// is used as the prefix of a Web address rather than the common http://, the session is managed by a security protocol, which is typically SSL, and the transmission is encrypted to and from the Web server.

- **(Secure Sockets Layer):** The leading security protocol on the Internet. Developed by Netscape, SSL is widely used to do two things: to validate the identity of a Web site and to create an encrypted connection for sending credit card and other personal data. Look for a lock icon at the top or bottom of your browser when you order merchandise on the Web. If the lock is closed, you are on a secure SSL or TLS connection.

- **A Stateless Connection:** HTTP is a "stateless" request/response system. The connection is maintained between client and server only for the immediate request, and the connection is closed. After the HTTP client establishes a TCP connection with the server and sends it a request command, the server sends back its response and closes the connection.

The first version of HTTP caused considerable overhead. Each time a graphics file on the page was requested, a new protocol connection had to be established between the browser and the server. In HTTP Version 1.1, multiple files could be downloaded with the same connection. It also improved caching and made it easier to create virtual hosts (multiple Web sites on the same server).

- **HTTP session :** An HTTP session is a sequence of network request-response transactions. An HTTP client initiates a request by establishing a Transmission Control

Protocol (TCP) connection to a particular port on a server. An HTTP server listening on that port waits for a client's request message. Upon receiving the request, the server sends back a status line, such as "HTTP/1.1 200 OK", and a message of its own, the body of which is perhaps the requested resource, an error message, or some other information.

2.6 DOMAIN NAME SYSTEM (DNS)

Domain **N**ame **S**ystem (or **S**ervice or **S**erver), an Internet service that translates domain names into IP addresses. Because domain namesare alphabetic, they're easier to remember. The Internet however, is really based on IP addresses. Every time you use a domain name, therefore, a DNS service must translate the name into the corresponding IP address. For example, the domain name www.example.com might translate to 198.105.232.4.

The DNS system is, in fact, its own network. If one DNS server doesn't know how to translate a particular domain name, it asks another one, and so on, until the correct IP address is returned.

Domain names are used to identify one or more IP addresses. For example, the domain name microsoft.com represents about a dozen IP addresses. Domain names are used in URLs to identify particular Web pages. For example, in the URL http://www.pcwebopedia.com/index.html, the domain name is pcwebopedia.com.

Some example Internet domain names:

1. about.com
2. navy.mil
3. harvard.edu
4. monster.ca
5. wikipedia.org
6. japantimes.co.jp
7. spain.info
8. sourceforge.net
9. wikipedia.org

Every domain name has a suffix that indicates which top level domain (TLD) it belongs to. There are only a limited number of such domains. For example:

- **gov -** Government agencies
- **edu -** Educational institutions
- **org -** Organizations (nonprofit)
- **mil -** Military
- **com -** commercial business
- **net -** Network organizations
- **ca -** Canada
- **th -** Thailand

Because the Internet is based on IP addresses, not domain names, every web server requires a Domain Name System (DNS) server to translate domain names into IP addresses.

How Domain Names Are Spelled

1. Domain names are organized right to left, with general descriptors to the right, and specific descriptors to the left. It is like family surnames to the right, specific person names to the left. These descriptors are called "domains".
2. The "top level domains" (TLD, or parent domain) is to the far right of a domain name. Mid level domains (children and grandchildren) are in the middle. The machine name, often "www", is to the far left.
3. Levels of domains are separated by periods ("dots").
    - Example 1 above) **About** is the mid-level domain, **.com** is the top level domain.
    - Example 7 above) j**apantimes** is the smaller mid-level domain. **.co** is the larger mid-level domain. **.jp** is the top level domain.
    - Example 10 above) **spain** is the mid-level domain, **.info** is the top level domain.

A Domain Name is Not the Same as URL

To be technically correct, a domain name is commonly part of a larger Internet address called a "URL". A URL goes into much more detail than domain name, providing much more information, including the specific page address, folder name, machine name, and protocol language.

Example Uniform Resource Locator pages, with their domain names bolded:

1. http://horses.**about.com**/od/basiccare/a/healthcheck.htm
2. http://www.**nytimes.com**/2007/07/19/books/19potter.html
3. http://www.**nrl.navy.mill**/content.php?P=MISSION

On the Internet, a domain consists of a set of network addresses. This domain is organized in levels. The top level identifies geographic or purpose commonality (for example, the nation that the domain covers or a category such as "commercial"). The second level identifies a unique place within the top level domain and is, in fact, equivalent to a unique address on the Internet (an IP address). Lower levels of domain may also be used.

A domain name is basically the name of your website such as google.comor cbc.ca or computercops.biz. These are all handled by companies known as domain name 'registrars' who submit entries to an international database overseen by the *Internet Corporation for Assigned Names and Numbers* (ICANN).

The domain name that you select basically consists of two parts:

- **Second level domain**-The actual name of your website,
- e.g.,ironspider
- **Top level domain (TLD)**-The 2 or 3 letter suffix at the end of the domain name, e.g. .com, .org, .ca, .uk, etcetera.

2.7 URLs

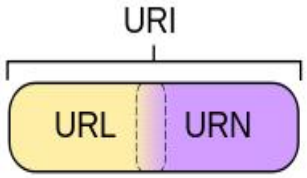


Fig. 2.4

Abbreviation of Uniform Resource Locator (URL) is the global address of documents and other resources on the World Wide Web. Uniform Resource Locator is the address that defines the route to a file on an Internet server (Web server, FTP server, mail server, etc.). URLs are typed into a Web browser to access Web pages and files, and URLs are embedded within the pages themselves as hypertext links.

A URL is technically a type of uniform resource identifier (URI) but in many technical documents and verbal discussions URL is often used as a synonym for URI. **U**niform **R**esource **I**dentifier is the addressing technology for identifying resources on the Internet or private intranet. URIs were originally defined as two types: Uniform Resource Locators (URLs) which are addresses with network locations, and Uniform Resource Names (URNs), which are persistent names that are address independent.

The terms URI and URL are used synonymously, but URL is more widely used in everyday conversation. In technical documentation from the standards committees, URI is the preferred term and may be used with the application-level protocol (URI scheme) such as HTTP to be more specific. Thus, an "HTTP URI" is a URL used to access a Web server.

The Components of a URL

A web URL follows a standard syntax that can be broken down into a few key parts, diagrammed in Figure 1. Each segment of the URL communicates specific information to both the client and the server.

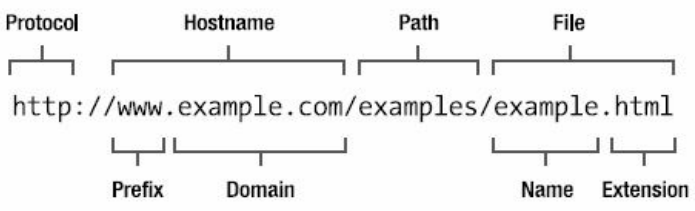


Fig. 2.5

- The protocol indicates one of a few different sets of rules that dictate the movement of data over the Internet. The web uses HyperText Transfer Protocol (HTTP), the standard protocol used for transmitting hypertext-encoded data from one computer to another.
- The protocol is separated from the rest of the URL by a colon and two forward slashes (://).
- A hostname is the name of the site from which the browser will retrieve the file. The web server's true address is a unique numeric Internet Protocol (IP) address, and every computer connected to the Internet has one. IP addresses look something like "65.19.150.101".
- A domain name is a more memorable alias that can be used to direct Internet traffic to an IP address.
- A prefix can be almost any short text label, but "www" is traditional. It's

possible for another entire website to exist separately within a domain under a different prefix, known as a subdomain.

- A hostname will also feature a domain suffix (sometimes called an extension) to indicate the category of domain the host resides in, such as ".com" for a U.S. commercial domain, ".edu" for a U.S. educational institution, or ".co.uk" for a commercial website in the United Kingdom.
- The path specifies the directory on the web server that holds the requested document, just as you probably save files in different virtual folders on your own computer.
- The specific file to retrieve is identified by its file name and extension. An HTML (or XHTML) document will have an extension of .html or .htm.

Absolute and Relative URLs

A URL can take either of two forms when it points to a resource elsewhere within the same site.

An ***absolute URL*** is one that includes the full string, including the protocol and hostname, leaving no question as to where that resource is found on the web. You'll use an absolute URL when you link to a site or file outside your own site's domain, though even internal URLs can be absolute.

```
http://www.computerlanguage.com/examples/chapter1/example.html
```

A ***relative URL*** is one that points to a resource within the same site by referencing only the path and/or file, omitting the protocol and hostname since those can be safely assumed. It might look something like this:

```
examples/chapter1/example.html
```

2.8 TCP/IP (TRANSMISSION CONTROL PROTOCOL / INTERNET PROTOCOL)

TCP/IP (Transmission Control Protocol/Internet Protocol) is the basic communication language or protocol of the Internet. It can also be used as a communications protocol in a private network (either an intranet or an extranet). When you are set up with direct access to the Internet, your computer is provided with a copy of the TCP/IP program just as every other computer that you may send messages to or get information from also has a copy of TCP/IP.

TCP/IP is a two-layer program. The higher layer, Transmission Control Protocol, manages the assembling of a message or file into smaller packets that are transmitted over the Internet and received by a TCP layer that reassembles the packets into the original message. The lower layer, Internet Protocol, handles the address part of each packet so that it gets to the right destination. Each gateway computer on the network checks this address to see where to forward the message. Even though some packets from the same message are routed differently than others, they'll be reassembled at the destination.

TCP/IP uses the client/server model of communication in which a computer user (a client) requests and is provided a service (such as sending a Web page) by another computer (a server) in the network. TCP/IP communication is primarily

point-to-point, meaning each communication is from one point (or host computer) in the network to another point or host computer. TCP/IP and the higher-level applications that use it are collectively said to be "stateless" because each client request is considered a new request unrelated to any previous one (unlike ordinary phone conversations that require a dedicated connection for the call duration). Being stateless frees network paths so that everyone can use them continuously. (Note that the TCP layer itself is not stateless as far as any one message is concerned. Its connection remains in place until all packets in a message have been received.)

Many Internet users are familiar with the even higher layer application protocols that use TCP/IP to get to the Internet. These include the World Wide Web's Hypertext Transfer Protocol (HTTP), the File Transfer Protocol (FTP), Telnet (Telnet) which lets you logon to remote computers, and the Simple Mail Transfer Protocol (SMTP). These and other protocols are often packaged together with TCP/IP as a "suite."

Personal computer users with an analog phone modem connection to the Internet usually get to the Internet through the Serial Line Internet Protocol (SLIP) or the Point-to-Point Protocol (PPP). These protocols encapsulate the IP packets so that they can be sent over the dial-up phone connection to an access provider's modem.

Protocols related to TCP/IP include the User Datagram Protocol (UDP), which is used instead of TCP for special purposes. Other protocols are used by network host computers for exchanging router information. These include the Internet Control Message Protocol (ICMP), the Interior Gateway Protocol (IGP), the Exterior Gateway Protocol (EGP), and the Border Gateway Protocol (BGP).

2.9 IP ADDRESS (INTERNET PROTOCOL ADDRESS)

The Internet Protocol (IP) is the method or protocol by which data is sent from one computer to another on the Internet. Each computer (known as a host) on the Internet has at least one IP address that uniquely identifies it from all other computers on the Internet.

When you send or receive data (for example, an e-mail note or a Web page), the message gets divided into little chunks called packets. Each of these packets contains both the sender's Internet address and the receiver's address. Any packet is sent first to a gateway computer that understands a small part of the Internet. The gateway computer reads the destination address and forwards the packet to an adjacent gateway that in turn reads the destination address and so forth across the Internet until one gateway recognizes the packet as belonging to a computer within its immediate neighborhood or domain. That gateway then forwards the packet directly to the computer whose address is specified.

Because a message is divided into a number of packets, each packet can, if necessary, be sent by a different route across the Internet. Packets can arrive in a different order than the order they were sent in. The Internet Protocol just delivers them. It's up to another protocol, the Transmission Control Protocol (TCP) to put them back in the right order.

IP is a connectionless protocol, which means that there is no continuing connection between the end points that are communicating. Each packet that

travels through the Internet is treated as an independent unit of data without any relation to any other unit of data. (The reason the packets do get put in the right order is because of TCP, the connection-oriented protocol that keeps track of the packet sequence in a message.) In the Open Systems Interconnection (OSI) communication model, IP is in layer 3, the Networking Layer.

The most widely used version of IP today is Internet Protocol Version 4 (Ipv4). However, IP Version 6 (IPv6) is also beginning to be supported. IPv6 provides for much longer addresses and therefore for the possibility of many more Internet users. IPv6 includes the capabilities of IPv4 and any server that can support IPv6 packets can also support IPv4 packets.

2.10 FILE TRANSFER PROTOCOL (FTP)

File Transfer Protocol (FTP) is a standard Internet protocol for transmitting files between computers on the Internet. Like the Hypertext Transfer Protocol (HTTP), which transfers displayable Web pages and related files, and the Simple Mail Transfer Protocol (SMTP), which transfers e-mail, FTP is an application protocol that uses the Internet's TCP/IPprotocols. FTP is commonly used to transfer Web page files from their creator to the computer that acts as their server for everyone on the Internet. It's also commonly used to download programs and other files to your computer from other servers.

As a user, you can use FTP with a simple command line interface (for example, from the Windows MS-DOS Prompt window) or with a commercial program that offers a graphical user interface. Your Web browser can also make FTP requests to download programs you select from a Web page. Using FTP, you can also update (delete, rename, move, and copy) files at a server. You need to logon to an FTP server. However, publicly available files are easily accessed using anonymous FTP.

Basic FTP support is usually provided as part of a suite of programs that come with TCP/IP. However, any FTP client program with a graphical user interface usually must be downloaded from the company that makes it.

FTP supports two modes of data transfer: plain text (ASCII), and binary. You set the mode in the FTP client. A common error when using FTP is attempting to transfer a binary file (such as a program or music file) while in text mode, causing the transferred file to be unusable.

2.10.1 How to use FTP

Graphical FTP clients : Graphical FTP clients simplify file transfers by allowing you to drag and drop file icons between windows. When you open the program, enter the name of the FTP host (e.g., `ftp.empire.gov`) and your username and password. If you are logging into an anonymous FTP server, you may not have to enter anything. Two common FTP programs are Cyberduck (for Mac) and WinSCP (for Windows).

Web browser

You can use a web browser to connect to FTP addresses exactly as you would to connect to HTTP addresses. Using a web browser for FTP transfers makes it easy

for you to browse large directories and read and retrieve files. Your web browser will also take care of some of the details of connecting to a site and transferring files. While this method is convenient, web browsers are often slower and less reliable and have fewer features than dedicated FTP clients.

To use your web browser to connect to an FTP site such as ftp.empire.gov, where you normally enter a URL, enter:

```
ftp://username@ftp.empire.gov/
```

2.10.2 FTP Client

An FTP Client is software that is designed to transfer files back-and-forth between two computers over the Internet. It needs to be installed on your computer and can only be used with a live connection to the Internet.

The classic FTP Client look is a two-pane design. The pane on the left displays the files on your computer and the pane on the right displays the files on the remote computer.

File transfers are as easy as dragging-and-dropping files from one pane to the other or by highlighting a file and clicking one of the direction arrows located between the panes. Additional features of the FTP Client include: multiple file transfer; the auto re-get or resuming feature; a queuing utility; the scheduling feature; an FTP find utility; a synchronize utility; and for the advanced user, a scripting utility.

Windows	**Mac**
• Filezilla	• Cyberduck
• SmartFTP	• gFTP
	• Transmit
Linux	**Other**
• KFTPGrabber	• Net2FTP
• kasablanca	• Fire FTP

Fig. 2.6

What are the available FTP programs?

You should be able to find a wide range of FTP programs by using a search engine, but here is a list of some that you can try:

2.11 SEARCHING TECHNIQUES

The Internet is a vast computer database. As such, its contents must be searched according to the rules of computer database searching. Search engines have a variety of ways for you to refine and control your searches. Some of them offer menu systems for this. Others require you to use special commands as part of your query.

Following are some searching and advance techniques to get the required information from the internet.

1. Boolean Logic : Boolean Logic is derived from a system of logic designed to produce better search results by formulating more precise queries. We use the Boolean logic operators: AND, OR, and NOT, to link words and phrases for more precise queries.

- **The AND operator:** Using AND between multiple search terms means that results must contain both terms in the document found for the search engine or database to retrieve it.
- **The OR operator:** Using OR between multiple terms means that any term can be present in the document.
- **The NOT operator:** Using NOT between keyword search terms limits your search results to records that have the first keyword but not the second keyword, even if the first word appears in that document, too.

2. Proximity Searching

- **The ADJ operator:** Using ADJ finds records that contain both search words adjacent to each other, in the exact order they were entered in the search box.
- **The NEAR operator:** NEAR finds records that contain both keywords right next to each other, in the same sentence, in either order.
- **The WITH Operator:** WITH helps locate records that contain both words in the same sentence. This is another tool to limit search results.

Additional Search Techniques

3. **Combining Search Operators :** You can combine different Boolean operators to create more focused results.
4. **Truncation and Word Stemming :** Word Stemming expands your search, but in a focused, useful way. Use an asterisk (*) or a dollar sign ($) at the end or in the middle of a full or partial word to etrieve all variants of that word in a document.
5. **Nesting :** Nesting terms will allow you to search for specific word variations with one search. Nesting expands the number of responses you will retrieve, but focuses the records retrieved at the same time.
6. **Case Sensitivity :** It is important to know that some web search tools are case sensitive. This means that when you search using upper case letters in your query, the search engine will only find results that are displayed in upper case letters.
7. **Phrase Searching :** Phrase searching retrieves a series of words (and only those words) in the exact order you type them. If any words are missing, not in the order typed, or have various spellings, the search will not retrieve them. Phrase searching greatly limits searches, but it is sometimes too strict of a search that misses many other valuable resources. Phrase searches must be enclosed in quotes for most search tools.

2.12 SEARCH ENGINE

A search engine is a software program that searches for sites based on the words that you designate as search terms. Search engines look through their own databases of information in order to find what it is that you are looking for. On the Internet, a search engine is a coordinated set of programs that includes:

- A spider (also called a **"crawler"** or a **"bot"**) that goes to every page or representative pages on every Web site that wants to be searchable and reads it, using hypertext links on each page to discover and read a site's other pages
- A program that creates a huge index (sometimes called a **"catalog**") from the pages that have been read
- A program that receives your search request, compares it to the entries in the index, and returns results to you

Search engines are the key to finding specific information on the vast expanse of the World Wide Web. Without sophisticated search engines, it would be virtually impossible to locate anything on the Web without knowing a specific URL. There are basically two types of search engines: Those that are powered by robots (called **crawlers; ants or spiders**) and those that are powered by human submissions.

- Crawler-based search engines are those that use automated software agents (called crawlers) that visit a Web site, read the information on the actual site, read the site's meta tags and also follow the links that the site connects to performing indexing on all linked Web sites as well. The crawler returns all that information back to a central depository, where the data is indexed. The crawler will periodically return to the sites to check for any information that has changed. The frequency with which this happens is determined by the administrators of the search engine.
- Human-powered search engines rely on humans to submit information that is subsequently indexed and catalogued. Only information that is submitted is put into the index.

2.12.1 Search Tools

There are three basic types of search tools that most people use to find what they are looking for on the Web (there's more than this, but these are the basics that everyone should start with.

- Search Engines
- Subject Directories
- MetaSearch Tools

Search the Web with Search Engines : Search engines are large, spider created databases of web pages that help searchers find specific information on any given subject. You type in a keyword or phrase and the search engine retrieves pages that correspond to your search query.

Search results gathered from these search engines are not always relevant to the keywords entered, since these engines are not intuitive and cannot infer dynamically what it is you might be searching for (although results are getting better all the time).

Interpretation of relevancy is different in each search engine. Many search engines have included categories to direct users to more relevant sites based on these

particular topics. Want to learn more about search engines? Check out my article titled or discover literally hundreds of search engines with the The Ultimate Search Engine List.

Search the Web with Subject Directories : Subject directories in general are more smaller and selective that search engines. They use categories to focus your search, and their sites are arranged by categories, not just by keywords. Subject directories are handy for broad searches, as well as finding specific web sites. Most subject directories' main purpose is to be informational, rather than commercial. A good example of a search directory is Yahoo, a combination search engine/search directory/ search portal, or one of the original search directories, Open Directory or DMOZ for short.

Search the Web with Meta search Engines : Meta search engines get their search results from several search engines. Users will receive the best hits to their keywords from each search engine. Meta search tools are a good place to start for very broad results, but do not (usually) give the same quality results as using each search engine and directory. Check out my list of profiled meta search engines.

2.12.2 How Do Search Engines Work?

Search engines are not simple. They include incredibly detailed processes and methodologies, and are updated all the time. This is a bare bones look at how search engines work to retrieve your search results. All search engines go by this basic process when conducting search processes, but because there are differences in search engines, there are bound to be different results depending on which engine you use.

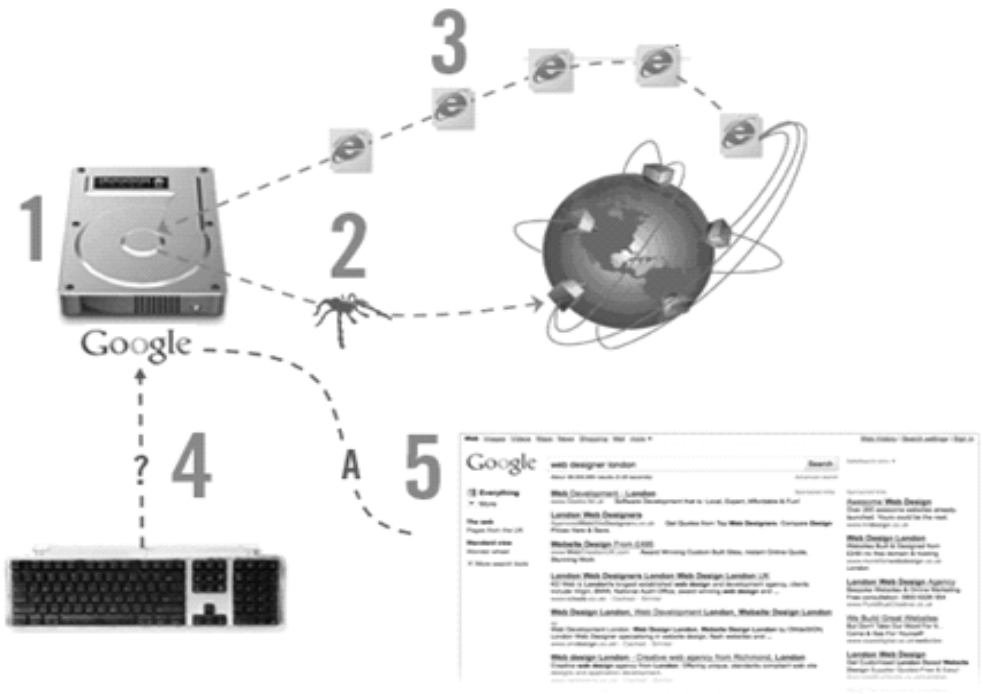


Fig. 2.7

- Search engines' servers have programs on their Web servers called “spiders."
- These spiders crawl the Web to gather page content.
- The spiders find pages on your website and collect information from the content and meta tags.
- The search engines' spiders follow the internal links on your site and index its content.
- When searchers search on a given keyword or phrase, results from the index are displayed in order of rank on the SERP.

2.13 WAP–INTRODUCTION

WAP is the de facto worldwide standard for providing Internet communications and advanced telephony services on digital mobile phones, pagers, personal digital assistants and other wireless terminals - WAP Forum

WAP stands for **W**ireless **A**pplication **P**rotocol. Per the dictionary definition for each of these words, we have:

- **Wireless:** Lacking or not requiring a wire or wires: pertaining to radio transmission.
- **Application:** A computer program or piece of computer software that is designed to do a specific task.
- **Protocol:** A set of technical rules about how information should be transmitted and received using computers.

WAP is the set of rules governing the transmission and reception of data by computer applications on, or via, wireless devices like mobile phones. WAP allows wireless devices to view specifically designed pages from the Internet, using only plain text and very simple black-and-white pictures.

WAP is a standardized technology for cross-platform, distributed computing, very similar to the Internet's combination of Hypertext Markup Language (HTML) and Hypertext Transfer Protocol (HTTP), except that it is optimized for:

- low-display capability
- low-memory
- low-bandwidth devices, such as personal digital assistants (PDAs), wireless phones, and pagers.

WAP is designed to scale across a broad range of wireless networks, like GSM, IS-95, IS-136 and PDC.

To browse a standard internet site you need a web browser. Similar way to browse a WAP enables website you would need a micro browser. A Micro Browser is a small piece of software that makes minimal demands on hardware, memory and CPU. It can display information written in a restricted mark-up language called WML. Although tiny in memory footprint, it supports many features and is even scriptable.

Today, all the WAP enabled mobile phones or PDAs are equipped with these micro browsers so that you can take full advantage of WAP technology.

2.13.1 Why is WAP Important?

Until the first WAP devices emerged, the Internet was the Internet and a mobile phone was a mobile phone. You could surf the Net, do serious research, or be entertained on the Internet using your computer. But this was limited to your computer.

Now with the appearance of WAP, the scene is that we have the massive information, communication, and data resources of the Internet becoming more easily available to anyone with a mobile phone or communications device.

WAP, being open and secure, is well suited for many different applications, including but not limited to stock market information, weather forecasts, enterprise data and games.

Despite the common misconception, developing WAP applications requires only a few modifications to existing web applications. The current set of web application development tools will easily support WAP development, and in the future more development tools will be announced.

2.13.2 WAP–Key Features

There are listed some of the key features offered by WAP:

A programming model similar to the Internet's: Though WAP is a new technology but it reuse the concepts found on the Internet. This reuse enables a quick introduction of WAP-based services, since both service developers and manufacturers are familiar with these concepts today.

Wireless Markup Language (WML): You must be using HTML language to develop your web based application. Same way WML is a markup language used for authoring WAP services, fulfilling the same purpose as HTML does on the Web. In contrast to HTML, WML is designed to fit small handheld devices.

WMLScript: Once again, you must be using Java Script or VB script to enhance the functionality of your web applications. Same way WMLScript can be used to enhance the functionality of a service, just as, Java script can be utilized in HTML. It makes it possible to add procedural logic and computational functions to WAPbased services.

Wireless Telephony Application Interface (WTAI): The WTAI is an application framework for telephony services. WTAI user agents are able to make calls and edit the phone book by calling special WMLScript functions or by accessing special URLs. If one writes WML decks containing names of people and their phone numbers, you may add them to your phone book or call them right away just by clicking the appropriate hyperlink on the screen.

Optimized protocol stack : The protocols used in WAP are based on well-known Internet protocols, such as HTTP and Transmission Control Protocol (TCP), but they have been optimized to address the constraints of a wireless environment, such as low bandwidth and high latency.

2.14 INTERNET SERVICES

As public libraries start to provide public access to the Internet they are

faced with a number of policy issues such as whether Internet workstations can be used for email, whether users can use chat rooms, or download material to a workstation's hard drive. This issue paper identifies some of the main types of Internet service available, the issues surrounding the provision of these services and the importance of having an acceptable use policy which explains why the library is willing to provide some services and not others.

Web Access: One of the main reasons most public libraries provide access to the Internet is to give their users access to another information resource. Provision of the opportunity to 'surf the net' is the baseline networked service. Even here there are choices to be made concerning charging, printing, downloading, filtering, and providing services such as links to selected sites and assistance or training for users.

Downloading: Downloading of material from the web can be done either to the computer's hard drive or to floppy disks. The former can be problematic for security reasons, because of the danger of viruses, and because many files take a long time to download and can thus tie up the computer for an unacceptable length of time.

E-mail: The Internet is not purely a source of information however. It is also a means of communication, and many users will want to use it to send and receive messages.

Allowing users to set up personal e-mail accounts (e.g. a.reader@countyshire.gov.uk) on library computers can be administratively and financially problematical and is seldom done in public libraries, although as networked services become more common and users' demands more sophisticated this situation may change. If people are prepared to pay for this sort of service it could become a good income generator.

Bulletin boards and mailing lists: These allow users to place messages on the Internet which are delivered to all subscribers via e-mail. Recipients can then reply in their own time. They can be a very useful source of information, providing access, for example, to a group of chrysanthemum growers who might provide the answer to a question not available in the gardening section of the library. Of course, as these require access to e-mail, they will be unavailable if e-mail is not a service offered.

Videoconferencing: This service is expensive to provide, but it has useful applications, for example, in connection with any lifelong learning services offered, as it could enable students to talk to their tutors and to each other. It can also be used to link with other services such as council tax or housing benefits departments and enable users to interact with advisers.

Telnet: Now that most resources can be viewed using a web browser, Telnet is not as essential as it once was for searching purposes. However, there are still some sites, mostly library catalogues as it happens, which can only be accessed via Telnet, and Telnet is also necessary for users who wish to use a library workstation to access e-mail on a server elsewhere. Offering Telnet is simply a matter of installing the necessary software, and is worth considering for these reasons.

E-commerce: This generally means buying and selling goods or services over the Internet, and might be regarded by some libraries as inappropriate use of a library's computer. However, some libraries might want to provide their users with access to

online bookshops; see Essex Libraries' online shopping arcade which links users to amazon.com and WH Smith where they can purchase books, music and videos. Business use corresponding to the type of service traditionally offered by library business information services is of course acceptable, and indeed desirable, contributing as it does to the local economy.

Interactive learning services: Public libraries have always been in the education business, and are increasingly offering flexible or open learning services as part of the lifelong learning agenda. Also, the University for Industry is making learning opportunities available through such venues as public libraries, and there is scope for partnerships with other educational providers, Training and Enterprise Councils, etc. It is eminently desirable therefore to consider offering access to online tutorials, not only on such subjects as how to use the Internet, but on a range of other subjects too. Links to suitable courses can be installed on the library's computers, and it is particularly useful to have Internet tutorials signposted from the home page for the benefit of novice users.

Social networking and entertainment: Social networking websites such as Facebook, Twitter, and MySpace have created new ways to socialize and interact. Users of these sites are able to add a wide variety of information to pages, to pursue common interests, and to connect with others. It is also possible to find existing acquaintances, to allow communication among existing groups of people. Sites like LinkedIn foster commercial and business connections. YouTube and Flickr specialize in users' videos and photographs.

2.15 STEPS TO CREATE YOUR WEB SITE

As you know, millions of web sites exist on the World Wide Web. You must be wondering how to have your own site, wherein you can display information of your company or even have your own personal information. Well, to establish a World Wide Web site, the following four components need to be considered:

- the connection
- server hardware
- server software
- content

If you plan to conduct financial transactions through your Web site, consider the Web server software that supports encryption to secure these transactions. You may decide whether to do the work involved in each component, or outsource all or part of the work. Usually the connection, the server hardware and the software is taken care by either your Internet service providers or the company that is hosting your page on the server. What you really need to take care of is the content. The content will project the image of your company to the entire world.

To create your own pages, just follow the following steps.

Step 1: Decide on the content of the web page: Content means whatever information and services you would like to have on your site. Your Web page can give access to databases, software scripts, and additional files. You can also offer

interactive options, such as a search capability and order forms.

Step 2: Develop the web page: Decide on the graphics that will go on your web page. Develop the graphic images in the format that web supports (usually *.gif or *.jpg) Once the text, graphics and other media objects are ready, develop the page using html language or by using a software package like Microsoft FrontPage or Netscape Composer.

Step 3: Test the web site before making it active on the web: It is very important to test all the web pages before you make them active on the Internet. Check for the content, accuracy of graphics, typographical errors, your address and phone numbers and the hyperlinks.

Step 4 : Register you page with InterNic: Big companies get registered with InterNic and obtain their own domain name. For example www.silverline.com, where Silverline is the company's name.

Step 5 : Promote your web site: Once your Web site is set up, you must promote it to make it visible to others. First register your site with different search engines. So if anyone searches for the services or products you offer, your web site appears as the search results. Get your web site listed in several newsgroups and mailing lists. Inform all your business clients and associates of your URL address through business cards, literature, and business meetings.

Step 6 : Enrich the content of your site: To make an efficient use of the site you have, attract attention of the visitors. If you want visitors to "PAY" attention to your site, then in return you must "GIVE' something to the visitors for paying this attention. From the web site, you can give additional interesting content to your target audience. Offer something free from your site, it could be free tickets, free software or something else.

Step 7: Update you site regularly: It is important to update information on your site frequently. The frequency of updating depends on the company. Remember, visitors do not re-visit sites that don't change for months at a time.

Step 8: Advertise your page: You have to pay for advertising your sites on other sites. Decide on your budget and target audience, before you decide where to advertise your site.

Step 9: Personal web sites or web sites for small companies: If you wish to put up your personal page or if you are working for a very small company, who would like to have presence on the web free of charge then search for the sites which offer you free hosting of your web pages. Some web sites host your page, free of cost. Actually, they advertise their sites through your page. These sites are good if you want to put personal information on the web. (tripod, geocities, rediffonthenet, etc.)

2.16 ENHANCING YOUR WEBSITE

Possession of appropriate tools or products is not enough to guarantee your website's success. There are other factors that should be factored in when building your website. Unfortunately, many of these factors are not put into consideration by internet business owners. The following is a guide of eight steps that could

enhance your website and its profitability:

1. Build It for Speed : These days, people have no time to waste reading detailed content. Therefore, you should capture the attention of your potential customers within 10 to 30 seconds. This you can achieve by reducing load time and ensuring that graphics remain small. **Flashy technology** such as Streaming Audio/Video, Flash, animation, JavaScript et cetera should be sparingly used and only used when it is very essential to your presentation.

2. Target your Market : You should know your market and alter the contents of your site to fit your customers' needs. It is very important for your website to be a reflection of your potential customers' values. For instance, if your market consists of business professionals, your site should be professional and clean. If you target teenagers and young adults, your website should be casual and relaxed. The essence here is to know your market and design your website with regard to their preferences.

3. Focus the Site : Your website should be in accordance with your goal, which is to sell. If your business is selling a variety of products, each page should focus on a particular product/service instead of concentrating them onto a single page. Subdomains will enable you to achieve this kind of focus.

4. Build Credibility : It is important that your customers are able to trust you, this way; you are assured of making sales. Assume authority in the field you have settled in. You should not only market but provide credible information as well. When you provide customers with clear and concise articles on your site's subject, they will consider you to be an expert with regard to your niche. Provide clients with a succinct privacy statement as a way of building credibility. A prominent link directing customers to your privacy statement should appear on each website's page as well as on any location where you ask customers to give their personal information. Reveal your identity by providing valid contact information like mailing address and phone number.

5. Keep Navigation Simple : Easy and intuitive navigation makes the navigation process smooth and simple hence, making it convenient for visitors. Visitors are most often in a hurry and lack the time to navigate through the entire website in search of what they need. Therefore, it would be wise to include catalog and search features of great power to your website.

6. Keep it Consistent : The design, look and feel of the website should be consistent. It is unpleasantly chocking and disturbing for a customer to feel that they are on a different site. Therefore, colors and themes should be consistent throughout the site.

7. Make your Site interactive and Personalized : Your website should be interactive by providing feedback forms and email forms that enable potential customers to ask questions with regard to certain products. Personalization as another key factor increases convenience and sales for your business. Personalization technology enables you to govern up-selling and cross-selling for customers buying online. In addition, it helps you to gain ideas of which products to up-sell and cross-sell. An instance is where a disc cleaner can be offered to a customer who purchases a CD player.

8. Content is King : Appropriate content increases sales. Ask yourself if your content

is sending the intended message to visitors, if it is compelling or whether it guides visitors through the process of sales. Let others critique, review and edit your content so that it delivers the desired message. Make sure that the spelling and grammar of your content is correct.

2.17 GOOD WEB DESIGN

Web design can be deceptively difficult, as it involves achieving a design that is both usable and pleasing, delivers information and builds brand, is technically sound and visually coherent.

Principles for good Web design:

1. Precedence (Guiding the Eye): Good Web design, perhaps even more than other type of design, is about information. One of the biggest tools in your arsenal to do this is precedence. When navigating a good design, the user should be led around the screen by the designer. I call this precedence, and it's about how much visual weight different parts of your design have.

To achieve precedence you have many tools at your disposal:

- **Position:** Where something is on a page clearly influences in what order the user sees it.
- **Color:** Using bold and subtle colors is a simple way to tell your user where to look.
- **Contrast:** Being different makes things stand out, while being the same makes them secondary.
- **Size:** Big takes precedence over little (unless everything is big, in which case little might stand out thanks to Contrast)
- **Design Elements:** if there is a gigantic arrow pointing at something, guess where the user will look?

2. Spacing: Spacing makes things clearer. In Web design there are three aspects of space that you should be considering:

- **Line Spacing:** When you lay text out, the space between the lines directly affects how readable it appears. Too little space makes it easy for your eye to spill over from one line to the next, too much space means that when you finish one line of text and go to the next your eye can get lost. So you need to find a happy medium.
- **Padding:** Generally speaking text should never touch other elements. Images, for example, should not be touching text, neither should borders or tables. Padding is the space between elements and text. The simple rule here is that you should always have space there.
- **White Space:** First of all, white space doesn't need to be white. The term simply refers to empty space on a page (or negative space as it's sometimes called). White space is used to give balance, proportion and contrast to a page. A lot of white space tends to make things seem more elegant and upmarket.

3. Navigation: One of the most frustrating experiences you can have on a Web site is being unable to figure out where to go or where you are. There are two aspects of navigation to keep in mind:

- **Navigation - Where can you go?:** There are a few commonsense rules to remember here. Buttons to travel around a site should be easy to find towards the top of the page and easy to identify. They should look like navigation buttons and be well described. The text of a button should be pretty clear as to where it's taking you. Aside from the common sense, it's also important to make navigation usable. For example, if you have a rollover sub-menu, ensuring a person can get to the sub-menu items without losing the rollover is important. Similarly changing the color or image on rollover is excellent feedback for a user.
- **Orientation -** Where are you now?: There are lots of ways you can orient a user so there is no excuse not to. In small sites, it might be just a matter of a big heading or a 'down' version of the appropriate button in your menu. In a larger site, you might make use of bread crumb trails, sub-headings and a site map for the truly lost.

4. Typography: Text is the most common element of design, so it's not surprising that a lot of thought has gone into it. It's important to consider things like:

- **Font Choices:** Different types of fonts say different things about a design. Some look modern, some look retro. Make sure you are using the right tool for the job.
- **Font sizes:** Years ago it was trendy to have really small text. Happily, these days people have started to realize that text is meant to be read, not just looked at. Make sure your text sizes are consistent, large enough to be read, and proportioned so that headings and sub-headings stand out appropriately.
- **Spacing:** As discussed above, spacing between lines and away from other objects is important to consider. You should also be thinking about spacing between letters, though on the Web this is of less importance, as you don't have that much control.
- **Line Length:** There is no hard and fast rule, but generally your lines of text shouldn't be too long. The longer they are, the harder they are to read. Small columns of text work much better (think about how a newspaper lays out text).
- **Color:** One of my worst habits is making low-contrast text. It looks good but doesn't read so well, unfortunately. Still, I seem to do it with every Web site design I've ever made, tsk tsk tsk.
- **Paragraphing:** Before I started designing, I loved to justify the text in everything. It made for nice edges on either side of my paragraphs. Unfortunately, justified text tends to create weird gaps between words where they have been auto-spaced. This isn't nice for your eye when reading, so stick to left-aligned unless you happen to have a magic body of text that happens to space out perfectly.

5. Consistency: Consistency means making everything match. Heading sizes, font choices, coloring, button styles, spacing, design elements, illustration styles, photo choices, etc. Everything should be themed to make your design coherent between pages and on the same page. Keeping your design consistent is about being professional. Inconsistencies in a design are like spelling mistakes in an essay. They just lower the perception of quality. Whatever your design looks like, keeping it consistent will always bring it up a notch. Even if it's a bad design, at least make it a consistent, bad design.

6. Alignment: Keeping things lined up is as important in Web design as it is in print design. That's not to say that everything should be in a straight line, but rather that you should go through and try to keep things consistently placed on a page. Aligning makes your design more ordered and digestible, as well as making it seem more polished.

2.18 HOME PAGE

The phrase "home page" can refer to several things in the internet. It can refer to the page that your web browser defaults to when you first start it up or it can refer to a personal or hobby page on the internet. Here, "home page" will refer to the starting page or the entry page to your website. The file name of this page on your website is typically called one of the following:

- index.htm
- index.html
- default.htm
- default.html

To explain, let's say for example the URL to your website is:

```
http://www.your_domain_name.com
```

Since there is no file name tacked on to the end of the URL, when you put this into the address bar of your web browser and hit Go, your browser is actually requesting the root directory of files that exists at this URL. Web hosts usually have their servers configured to search for and open the index.htm or the index.html file when any directory is requested by a web browser. Although this helps to abbreviate some URLs by dropping the file name, its real purpose is to prevent visitors from getting a directory list of files when using a URL that requests a directory instead of a specific file.

Hence, the starting or entry page to your website, a.k.a. home page, should always be named index.htm or index.html and it should always be kept in the root directory that your web host has created for your HTML files.

A **home page** or **homepage** has various related meanings to do with web sites:

- It most often refers to the initial or main web page of a web site, sometimes called the **front page** (by analogy with newspapers).
- The web page or local file that automatically loads when a web browser starts or when the browser's "home" button is pressed; this is also called

a **start page**. The user can specify the URL of the page to be loaded, or alternatively choose e.g. to re-load the most recent web page browsed.

- A personal web page, for example at a web hosting service or a university web site, that typically is stored in the home directory of the user.
- In the 1990s the term was also used to refer to a whole web site, particularly a personal web site (perhaps because simple web sites often consisted of just one web page).

A **home page** can also be used outside the context of web sites, such as to refer to the principal screen of a user interface, which is also referred to as a **homescreen** on mobile devices.

2.19 WEB DESIGNING

Web design is the creation of Web pages and sites using HTML, CSS, JavaScript and other Web languages. Web design is just like design in general: it is the combination of lines, shapes, texture, and color to create an aesthetically pleasing or striking look. Web design is the work of creating design for Web pages.

The process of designing Web pages, Web sites, Web applications or multimedia for the Web may utilize multiple disciplines, such as animation, authoring, communication design, corporate identity, graphic design, human-computer interaction, information architecture, interaction design, marketing, photography, search engine optimization and typography.

- Markup languages (such as HTML, XHTML and XML)
- Style sheet languages (such as CSS and XSL)
- Client-side scripting (such as JavaScript and VBScript)
- Server-side scripting (such as PHP and ASP)
- Database technologies (such as MySQL)
- Multimedia technologies (such as Flash and Silverlight)

Web pages and Web sites can be static pages, or can be programmed to be dynamic pages that automatically adapt content or visual appearance depending on a variety of factors, such as input from the end-user, input from the Webmaster or changes in the computing environment (such as the site's associated database having been modified).

2.20 DYNAMIC HTML

DHTML stands for Dynamic HTML. It is NOT a language or a web standard. DHTML is the art of combining HTML, JavaScript, DOM, and CSS. According to the World Wide Web Consortium (W3C):

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

DHTML Technologies : Below is a listing of DHTML technologies:

- **HTML 4.0 :** The HTML 4.0 standard has rich support for dynamic content like:

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

DHTML is about using these features to create dynamic and interactive web pages.

- **JavaScript :** JavaScript is the scripting standard for HTML.

  DHTML is about using JavaScript to control, access and manipulate HTML elements.

- **HTML DOM :** The HTML DOM is the W3C standard Document Object Model for HTML. It defines a standard set of objects for HTML, and a standard way to access and manipulate them.

  DHTML is about using the DOM to access and manipulate HTML elements.

- **CSS :** CSS is the W3C standard style and layout model for HTML. It allows web developers to control the style and layout of web pages. HTML 4 allows dynamic changes to CSS.

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

2.21 WEB PUBLISHING PROCESS

Web publishing, or "online publishing," is the process of publishing content on the Internet. It includes creating and uploading websites, updating web pages, and posting blogs online. The published content may include text, images, videos, and other types of media.

In order to publish content on the web, you need three things:

1) Web development software

2) An Internet connection

3) A web server.

The software may be a professional web design program like Dreamweaver or a simple web-based interface like WordPress. The Internet connection serves as the medium for uploading the content to the web server. Large sites may use a dedicated web host, but many smaller sites often reside on shared servers, which host multiple websites. Most blogs are published on public web servers through a free service like Blogger.

Since web publishing doesn't require physical materials such as paper and ink, it costs almost nothing to publish content on the web. Therefore, anyone with the three requirements above can be a web publisher. Additionally, the audience is limitless since content posted on the web can be viewed by anyone in the world with an Internet connection. These advantages of web publishing have led to a new era of personal publishing that was not possible before.

Publishing Process : The three essential steps to publish your website onto the web are:

1. Creating your website: First you have to create a website. Depending on your level on knowledge in web development you have the following options. Either you can create the site yourself through manual programming in an editor program like DreamWeaver, EditPlus or Notepad. Otherwise there are a number of open source alternatives available without any requirement of prior programming knowledge.

2. Get a hosting plan: Next step is to find somewhere online to put your website files and assign an address to that somewhere. Any given hosting plan include storage space on a web server and a domain name registration. Depending on what size your website has and potential traffic volume it's going to generate, you get a hosting plan based on three factors - storage space, bandwidth and CPU.

3. Upload your files/Publish your website: Upload your files to the root directory of your purchased web hosting server space. You can do this either by using your hosting account's inhouse file manager, but those are often slow and manage large file volume poorly. Our recommendation is to get your hands on a FTP client.

Assuming that are new to site building and all that web developments entiles, for you to get a website up and running sooner rather than later, we suggest you follow the steps above. Use open source and the blessing of freeware. Get a hosting plan that matches your needs and publish your files through a FTP client like FireFtp. There will be things to remember and learn along the way, but if you follow this path, or these guidelines, creating a website will not seem so impossible.

2.22 PHASES OF WEBSITE DEVELOPMENT

There are numerous steps in the web site design and development process. From gathering initial information, to the creation of your web site, and finally to maintenance to keep your web site up to date and current. The exact process will vary slightly from designer to designer, but the basics are generally the same as:

- Information Gathering
- Planning
- Design
- Development
- Testing and Delivery
- Maintenance

Information Gathering : The first step in designing a successful web site is to gather information. Many things need to be taken into consideration when the look and feel of your site is created. This first step is actually the most important one, as it involves a solid understanding of the company it is created for. It involves a good understanding of you - what your business goals and dreams are, and how the web can be utilized to help you achieve those goals. Certain things to consider are:

- **Purpose:** What is the purpose of the site? Do you want to provide information, promote a service, sell a product ?
- **Goals:** What do you hope to accomplish by building this web site? Two of the more common goals are either to make money or share information.
- **Target Audience:** Is there a specific group of people that will help you

reach your goals? It is helpful to picture the "ideal" person you want to visit your web site. Consider their age, sex or interests - this will later help determine the best design style for your site.

- **Content:** What kind of information will the target audience be looking for on your site? Are they looking for specific information, a particular product or service, online ordering…?

Planning : Using the information gathered from phase one, it is time to put together a plan for your web site. This is the point where a site map is developed.

The site map is a list of all main topic areas of the site, as well as sub-topics, if applicable. This serves as a guide as to what content will be on the site, and is essential to developing a consistent, easy to understand navigational system. A good user interface creates an easy to navigate web site, and is the basis for this.

Designing : Drawing from the information gathered up to this point, it's time to determine the look and feel of your site. Target audience is one of the key factors taken into consideration. A site aimed at teenagers, for example, will look much different than one meant for a financial institution. As part of the design phase, it is also important to incorporate elements such as the company logo or colors to help strengthen the identity of your company on the web site.

Your web designer will create one or more prototype designs for your web site. This is typically a .jpg image of what the final design will look like. Often times you will be sent an email with the mock-ups for your web site, while other designers take it a step further by giving you access to a secure area of their web site meant for customers to view work in progress.

In this phase, communication between both you and your designer is crucial to ensure that the final web site will match your needs and taste. It is important that you work closely with your designer, exchanging ideas, until you arrive at the final design for your web site.

Development : The developmental stage is the point where the web site itself is created. At this time, your web designer will take all of the individual graphic elements from the prototype and use them to create the actual, functional site.

This is typically done by first developing the home page, followed by a "shell" for the interior pages. The shell serves as a template for the content pages of your site, as it contains the main navigational structure for the web site. Once the shell has been created, your designer will take your content and distribute it throughout the site, in the appropriate areas.

Elements such as interactive contact forms, flash animations or ecommerce shopping carts are implemented and made functional during this phase, as well.

This involves writing valid XHTML / CSS code that complies to current web standards, maximizing functionality, as well as accessibility for as large an audience as possible.

Testing : At this point, your web designer will attend to the final details and test your web site. They will test things such as the complete functionality of forms or other scripts, as well last testing for last minute compatibility issues (viewing differences between different web browsers), ensuring that your web site is optimized

to be viewed properly in the most recent browser versions.

A good web designer is one who is well versed in current standards for web site design and development. The basic technologies currently used are XHTML and CSS (Cascading Style Sheets). As part of testing, your designer should check to be sure that all of the code written for your web site validates. Valid code means that your site meets the current web development standards - this is helpful when checking for issues such as cross-browser compatibility as mentioned above.

Once you give your web designer final approval, it is time to deliver the site. An FTP (File Transfer Protocol) program is used to upload the web site files to your server. Most web designers offer domain name registration and web hosting services as well. Once these accounts have been setup, and your web site uploaded to the server, the site should be put through one last run-through. This is just precautionary, to confirm that all files have been uploaded correctly, and that the site continues to be fully functional.

This marks the official launch of your site, as it is now viewable to the public.

Maintenance : The development of your web site is not necessarily over, though. One way to bring repeat visitors to your site is to offer new content or products on a regular basis. Most web designers will be more than happy to continue working together with you, to update the information on your web site. Many designers offer maintenance packages at reduced rates, based on how often you anticipate making changes or additions to your web site.

If you prefer to be more hands on, and update your own content, there is something called a CMS (Content Management System) that can be implemented to your web site. This is something that would be decided upon during the Planning stage. With a CMS, your designer will utilize online software to develop a database driven site for you.

A web site driven by a CMS gives you the ability to edit the content areas of the web site yourself. You are given access to a back-end administrative area, where you can use an online text editor (similar to a mini version of Microsoft Word). You'll be able to edit existing content this way, or if you are feeling more adventurous, you can even add new pages and content yourself. The possibilities are endless.

2.23 WEBCASTING

The term "webcasting" holds a number of different meanings within modern communications. In its most basic form, the term webcasting simply refers to audio or video which is broadcast over the World Wide Web using a single content source to distribute to a wide trainee. This can be delivered either live, or "on demand" where the viewer has the flexibility to view whenever they wish. However, rather than requiring a download like a podcast or video podcast, a webcast uses a progressive video stream onto the users computer so there is no need for hard drive space or leftover media files.

Due to its generally accepted use, webcasting is most broadly used by the media to broadcast non-interactive entertainment and news over the web. The

main benefit of this is the flexibility provided to the viewer who is not constrained by time schedules like those of more traditional broadcasting. Within the UK, major television networks such as the BBC, ITV and Channel 4 all webcast their content over the web, both as a live stream and as an "on demand/watch again" service where all content is available for one week after its live broadcast. Although these webcasts are generally non-interactive, an interactive element is delivered through forums and chat rooms linked to the page supporting the webcast. This either allows viewers to discuss the content as an online community or, in some cases of live discussion and panel shows, contribute to the show directly by asking questions or making statements which can be relayed directly to the panel.

2.11.1 Webcasting Techniques

Podcast : A podcast is a unique type of webcast of audio and video files that are automatically delivered to the user's computer. The podcast resides at a web location with a unique feed address. Users subscribe to the podcast by submitting the feed address to an aggregator. New content is delivered automatically, whenever it becomes available, to all subscribers. Podcast are not real-time feeds, rather information is pre-recorded and posted, allowing users to access content off-line, at their convenience. Unlike a webcast, which directly downloads or streams content, a podcast is syndicated and requires an intermediary, aggregator, or feed reader to deliver the content.

RSS Feed : An RSS feed is a format that is used to publish text, audio, or video content to a user's computer or mobile device. Content that is usually streamed includes podcasts, news, blogs, or other frequently updated material, which is either summarized or delivered in full text. RSS, which stands for Really Simple Syndication, makes it easy for publishers to deliver regular updates of web-based content, and for users to stay current and keep track of a large number of favorite web sites or blogs. For advertisers, RSS feeds eliminate the problems traditional marketing channels encounter, including spam filters, delayed distribution, and search engine rankings. RSS content is read using one of almost 2,000 feed reading applications. Some readers work exclusively on mobile devices. A user subscribes to a feed by entering the feed's link into the reader or by clicking an RSS icon, often called a "chicklet" in a browser window that initiates the subscription process.

The reader checks the user's subscribed feeds regularly for new content, downloading any updates that it finds. A marketer uses an RSS feed as the delivery method for podcasts or webcasts.

Email Blast : An email blast uses electronic mail as a way to communicate with multiple recipients, and inform them of announcements, events, or offers. Companies send most email blasts to current or potential customers in the hopes of enhancing the relationship and increasing customer loyalty. Email blasts can also encourage a predetermined behavior, such as making an immediate purchase, signing up for a seminar, downloading a document, or registering for an event.

Email blasts have become a popular and effective way to reach users for several reasons. First, companies are able to distribute information to a wide range of

targeted customers at a relatively low cost. Marketers can accurately track results and user actions via web bugs, bounce messages, un-subscribes, read-receipts, and click-throughs. Results are immediate-users get information seconds after the email is sent, shortening the time-to-market of most promotions. Email marketing is second only to search marketing as the most effective, affordable online marketing tactic.

Flash Movie : Flash is a technology platform owned by Adobe Systems, that includes an integrated development environment player and application files, and has facilitated the spread of animation and rich interactive applications on the Internet. Flash Player is the client application that acts as a virtual machine that runs Flash files, and has become a popular method for adding animation and interactivity to web pages. The ubiquity of the Flash Player means that almost every browser-and every user-has access to movies, games, and videos. Adobe claims Flash reaches 97.3 percent of desktop Internet users. Companies have adopted Flash as a way to deliver animations, video, and interactivity in advertising, web sites, and emails. Flash demos allow users to see "live" demos of products over the web.

MULTIPLE CHOICE QUESTIONS

1. What is the full form of HTTP?

a. Hyper text transfer protocol
b. Hyper text transfer package
c. Hyphenation text test program
d. none of the above

2. What is a search engine?

a. a program that searches engines
b. a web site that searches anything
c. a hardware component
d. a machinery engine that search data

3. Which of the following protocol is not used in the Internet.

a. Telnet
b. WIRL
c. HTTP
d. Gopher

4. Who invented World Wide Web (WWW)?

a. Blaise Pascal
b. Charles Babbage
c. Herman Hollerith
b. Tim Berners-Lee

5. In HTML, Uniform Resource Identifier (URI) is used to

a. To create a frame document .
b. To create a image map in the webpage.
c. To customize the image in the webpage.
d. To identify a name or a resource on the internet.

6. A homepage is _____

a. an index of encyclopedia articles
b. where all Internet data is stored
c. required for access to the Internet
d. the first page of a website

7. Which of the following is used to explore the Internet?

a. Browser b. Spreadsheet

c. Clipboard d. Draw

8. What is Internet Explorer?

a. An Icon b. A File Manager

c. A Browser d. The Internet

9. What do I need to get onto the Internet?

a. Computer b. Modem

c. Browser d. All of the above

10. What is an ISP?

a. Internet System Protocol b. Internal System Program

c. Internet Service Provider d. None of the above

11. Which of the following is valid IP address?

a. 984.12.787.76 b. 192.168.321.10

c. 1.888.234.3456 d. 192.168.56.115

12. Which is not a domain name extension

a. .mil b. .org

c. .int d. .com

13. What is a FTP program used for?

a. Transfer files to and from an Internet Server

c. Designing a website

c. Connecting to the internet

d. None of the above

14. Which of the following are commonly found on web pages?

a. internet b. hyperlinks

c. intranet d. all of the above

15. "Yahoo", "Infoseek" and "Lycos" are _____?

a. Search Engines b. Browsers

c. News groups d. None of the above

16. What does the .com domain represents?

a. Education domain b. Commercial domain

c. Network d. None of the above

17. In Satellite based communication, VSAT stands for?

a. Very Small Aperture Terminal b. Varying Size Aperture Terminal

c. Very Small Analog Terminal d. None of the above

18. Outlook Express is a _____

a. E-Mail Client b. Browser

c. Search Engine d. None of the above

KEY TO OBJECTIVE QUESTIONS

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

SHORT QUESTION ANSWER

Q. 1 What is Intranet?

Ans. An intranet is a local area network(LAN), which may not be connected to the Internet but which has similar functions.

Q.2 Explain HTML (Hypertext Markup Language)?

Ans. HTML stands for Hypertext Markup Language. This is the standard method of publishing web documents onto the World Wide Web (WWW). HTML consists of tags surrounded by brackets.

Q.3 What is IP Address (Internet Protocol Address)?

Ans. Each computer is assigned an IP address. These are similar to phone numbers. When you attempt to connect to an IP address, you will connect to the computer with that IP address.

Q.4 What is E-mail?

Ans. E-mail stands for electronic mail. Most networks support some form of email. The most popular, of course, is Internet email. E-mail allows you to send text (such as a letter) to another person on another computer. In order to send an email, you have to know the email address of the recipient. Internet email addresses always start with the user's account name, then the at sign (@), then the name of the computer where the user gets his or her email. You can never have spaces in email or Web addresses. For example, my email address is: w@wdell.com

Q.5 Explain Firewall?

Ans. A firewall is a hardware and/or software boundary that prevents unauthorized users from accessing restricted files on a network.

Q.6 What is ISDN (Integrated Services Digital Network)?

Ans. Integrated Services Digital Network (ISDN) combines digital network services and voice into one. Users can access digital services at 115,200 bps.

Q.7 What is MIME (Multipurpose Internet Mail Extensions)?

Ans. Multipurpose Internet Mail Extensions, or MIME, is the standard way to organize different file formats. For example, if you receive an e-mail, which is in a different format than yours, the file will be decoded so you can read it using MIME.

Q.8 What is TCP / IP (Transmission Control Protocol / Internet Protocol)?

Ans. Transmission Control Protocol/Internet Protocol, or TCP/IP, is the basic communications protocol required for computers that use the Internet.

Q.9 What is Web Browser?

Ans. A web browser is a program that you use to view web pages. The two most popular web browsers are Microsoft Internet Explorer and Netscape Navigator.

Q.10 What is Cookies?

Ans. Provide a simple way to identify session among a group of HTTP/HTML requests. The cookie value is often an index into a table stored in the memory of a Web server that points to an in memory object holding the user's records. This has many potential problems: If the user's request is routed to a different server in a subwequent request, the session information is unknown to the server. If the user is rounted to a different server and the server is part of an application cluster, then all the servers that could receive the user's request must have a way to synchronize the session data. Storing cookies and synchronizing sessions among clusters of server usually requires configuration, storage space, and memory.

Q.11 What is DNS (Domain Name Service)?

Ans. A name service used with TCP/IP hosts. A DNS exists on numerous servers over the Internet. It is a database for finding host names and IP addresses on the Internet and trying to figure them out.

Q.12 What is Gopher?

Ans. A search and retrieval tool for information used mostly for research.

Q.13 What is the IP (Internet Protocol)?

Ans. A packet switching protocol that is used as a network layer in the TCP/IP protocol suite.

Q.14 What is ISP (Internet Service Provider)?

Ans. An organization or company that has a network with a direct link to the Internet. This is done by using a dedicated line connection, usually through a link known as a T1 connection. Users can dial into to that network using their modem. Most ISP's now charge a monthly fee.

Q.15 What is LAN (Local Area Network)?

Ans. Local Area Network. A LAN allows users to share files between computers, send e-mail and access the Internet. Most companies use Local Area Networks so that users can access information within or outside the LAN.

Q.16 What is POP (Post Office Protocol)?

Ans. A protocol that allows single users to read mail from a server.

Q.17 What is a Protocol?

Ans. A protocol is a method of communication between two devices. You can think of it as the language the devices use to communicate with each other, although it is not the same as a programming language (by which a human programmer controls a

computer). Different brands of printers, for example, each use their own protocol (or "language") by which a computer can communicate with the printer. This is why a driver program must be written for each printer.

Q.18 Explain SMTP (Simple Mail Transfer Protocol)?

Ans. A standard protocol used to transfer e-mail messages.

Q.19 Explain Telnet?

Ans. This is the standard Internet protocol to connect to remote terminals.

Q.20 What is URL (Universal Resource Locator)?

Ans. A Universal Resource Locator refers to the universal address of an Internet web page. A URL consists of three things. First, it starts with letters such as http, ftp, or gopher that identify the resource type, followed by a colon and two forward slashes. Next, the computer's name is listed. And finally, the filename and directory of the remote resource is listed as well.

Q.21 What is Web Page?

Ans. A web page is a rich document that can contain richly formatted text, graphics, animation, sound, and much more. Some web pages are generated dynamically (such as the results of a search). You are currently viewing a (static) web page. Every web page on the Internet has a unique address which starts with the name of the computer that holds that page. Within a web page, words and pictures can be linked to other pages. When you activate a link, you will be taken to another page automatically. See also: Web, Web Browser, Understanding Internet Addresses.

Q.22 Explain Web server?

Ans. A Web server is a server on the Internet that holds Web documents and makes them available for viewing by remote browsers.

****

INFORMATION ARCHITECTURE

IN THIS CHAPTER, YOU WILL LEARN

- Introduction to Information Architecture
- The Role of the information architect
- Collaboration and Communication
- Organization Information
- Challenges of Organizing Information
- Organizing Web Sites and Intranets

3.1 INTRODUCTION TO INFORMATION ARCHITECTURE

Each building serves its purpose uniquely. Architecture and design of a building depends upon the purpose, construction, users, finance etc. if we start constructing building without deciding the architecture then constructor will have problems in constructing, its user will have problems in using it and the purpose for which the building were constructed will never achieved. Similarly websites are resources of information. Each website serves its purpose uniquely. If websites is developed without any planning about design and architecture then developer may have problems in using the websites, searching and accessing the information. These problems may be like time consuming search, time wastage in loading of web page due to improper formula used and difficulty in browsing due to use of improper keywords, so information architecture is necessary.

A. For producer, so that any updation in the information can be done efficiently within time.

B. If users are facing difficulty in searching and navigation the information then they will not use the website again.

C. As unorganized information can't be converted into knowledge.

Definition of Information Architecture

1) The combination of organizing, labeling and navigation schemes within the information system.
2) The structural design of information space to facilitate task completion and inductive access to content.
3) The art of science of structuring and classification web sites and intranets to help people find and manage information.
4) An emerging discipline and community of practice focused on principle of design and architecture to the digital and.

Basic concepts of Information Architecture

Information: The term information is used to distinguish information architecture from data management and knowledge management. Data is fact and figures. Information is meaningful data or managed data. Knowledge is managed information. It is the stuff in people's head. Intelligent is the managed knowledge.

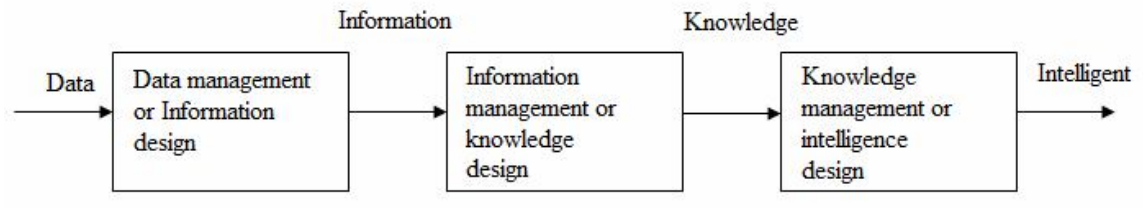


Fig. 3.1

Data manager produce highly structured database to produce specific answer to specific question. Knowledge managers develop tools, processes to encourage peoples to share knowledge. Information manager organize information atoms in different categories and label these categories to make it easy to understand.

Organizing information: It involves following steps

1. **Structuring:** Structuring information means determining appropriate levels of granularity for information atoms in your site and deciding how to relate them to another.
2. **Grouping:** Grouping information means grouping the linked information atoms into meaningful and distinctive categories.
3. **Labeling:** Labeling information means figuring out what to call these categories.
4. **Finding and managing:** Finding is need of users. Managing is goal of business.
5. **Art and Science:** Information Architecture is both art and science. Information Architecture must rely on experience, intuition and creativity, willing to take risks. This is the art of information Architecture. Information Architecture must follow the rules for organizing the information space. This is the science of information Architecture.

Components of information Architecture

1. **Organizing System:** How do we categorize Information?
2. **Labeling System:** How do we represent information?
3. **Navigation System:** How do we browse information?
4. **Searching System:** How do we search information?

Advantages of Information Architecture

- User gets a quick response.
- User can search easily.
- Maintenance of Information space becomes easy for the producer.

Disadvantages of Information Architecture

- Process of designing information Architecture is time consuming.

Definition Information Architect

1. The individual who organizes the patterns inherent in data, making the complex clear.
2. A person who creates the structure or map of information which allows others to find their personal paths to knowledge.

3. The emerging 21st century professional occupation addressing the needs of the age focused upon clarity, human understanding and the science of the organization of information.

3.2 THE ROLE OF THE INFORMATION ARCHITECT

The main jobs of the information architect are given below. An information Architect

- Clarifies the mission and vision for the site, balancing the needs of its sponsoring organization and the needs of its audiences.
- Determines what content and functionality the site will contain.
- Specifies how users will find information in the site by defining its organization, navigation, labeling, and searching systems.
- Maps out how the site will accommodate change and growth over time.

3.2.1 The Consumer's Perspective

Users want to find information quickly and easily. Poor information architectures make busy users confused, frustrated, and angry. Because different users have varying needs, it's important to support multiple modes of finding information. From consumer's perspective there can be two modes of finding information:

i) Known-item searching: Some users know exactly what they're looking for. They know what it's called (or labeled), and they know it exists. This is called known-item searching.

ii) Casual Browsing: Some users don't know what they are looking for. They may not know the right label. They casually explore your site and they have never even considered.

If you are care about the consumer make sure that your Information Architecture supports both modes. While attractive graphics and reliable technology are essential to user satisfaction, they are not enough.

3.2.2 The Producer Perspective

If you are producing an external web site, the user can be actual or perspective customers, investors, employees, business partner, media and senior executive. If you're producing an intranet, the employees of your organization are the consumers. The cost of designing and implementing the Information Architecture is the cost of time spent:

i) In designing categories of various users.

ii) In arguing over the main area of content and functionality that the site would include.

iii) Redesigning

iv) In maintaining the information space on increasing in information.

The role of Information Architect is to minimize their cost. If the information architect doesn't take care of producer perspective the burden would be on the

site's users to understand how to use and find information in a confusing, poorly-designed web site. The site's maintainers wouldn't know where to locate the new information that the site would eventually include, they had likely begin to quarrel over whose content was more important and deserved visibility on the main page, and so on.

3.2.3 Who Should Be the Information Architect ?

a) An insider who can understand the site's sponsoring organization, its main goals, audiences, and inner working.

Advantages

- Organization's information is in the safe hands
- No extra cost, so cost effective
- Insider knows the most about an organization process and how to get things done within the organization.

Disadvantages

- Knowledge of an insider may be too specific
- Insider may lack the political base require too mobilize corporation from the others in the organization
- Insider gets diverted from his original duties

b) Someone who can think as an outsider and be sensitive to needs of site user's

Advantages

- No biased behavior is expected from an outsider
- We have a choice for outsider, so we will choose according to our needs and he will act more efficiently than insider as he will be specialist of her field.

Disadvantages

- Extra cost
- Outsider doesn't have minute detail of the organization, so he needs information
- Passing secret information of the organization to an outsider can be dangers.

Outsider can be from a verity of field like:

i) **Journalism:** Journalists are good at editing and organizing information. They have rich knowledge base.

ii) **Graphic Design:** Graphic Design is much more than creating pretty pictures. It is geared more toward creating relationships between visual elements and determining their effective integration as a whole.

iii) **Information and library science:** People from this background are good to work with searching, browsing, and indexing technologies.

iv) **Marketing:** Marketing specialists are expert at understanding audiences and

communicating a message effectively to different audiences. They know how to highlight a positive feature and how to suppress the negative ones.

v) **Computer science:** Programmers and computer specialists bring an important skill to information architecture. Especially to architecting information from the bottom up. For example, often a site requires a database to serve the content; this minimizes maintenance and data integrity problems. Computer scientists have the best skills for modeling content for inclusion in a database.

3.3 COLLABORATION AND COMMUNICATION

The information architect must communicate effectively with the web site development team. This is challenging, since information architecture is highly abstract and intangible. Besides communicating the architecture verbally, documents (such as blueprint diagrams) must be created in ways that can be understood by the rest of the team regardless of their own disciplinary backgrounds.

In the early days of the Web, web sites were often designed, built, and managed by a single individual through sheer force of will. This webmaster was responsible for assembling and organizing the content, designing the graphics, and hacking together any necessary CGI scripts. The only prerequisites were a familiarity with HTML and a willingness to learn on the job. People with an amazing diversity of backgrounds suddenly became webmasters overnight, and soon found themselves torn in many directions at once. One minute they were information architects, then graphic designers, then editors, then programmers.

Then companies began to demand more of their sites and, consequently, of their webmasters. Simple home pages quickly evolved into complex web sites. People wanted more content, better organization, greater function, and prettier graphics. Extensions, plug-ins, and languages proliferated. Tables, VRML, frames, Shockwave, Java, and ActiveX were added to the toolbox. No mortal webmaster could keep up with the rising expectations and the increasing complexity of the environment.

Increasingly, webmasters and their employers began to realize that the successful design and production of complex web sites requires an interdisciplinary team approach. An individual cannot be an expert in all facets of the process. Rather, a team of individuals with complementary areas of expertise must work together. The composition of this team will vary, depending upon the needs of a particular project, available budget, and the availability of expertise.

However, most projects will require expertise in marketing, information architecture, graphic design, writing and editing, programming, and project management.

Marketing : The marketing team focuses on the intended purposes and audiences for the web site. They must understand what will bring the right people to the web site and what will bring them back again.

Information Architecture : The information architects focus on the design of organization, indexing, labeling, and navigation systems to support browsing and searching throughout the web site.

Graphic Design : The designers are responsible for the graphic design and page layout that defines the graphic identity or look of the web site. They strive to create and implement a design philosophy that balances form and function.

Editorial : Editors focus on the use of language throughout the web site. Their tasks may involve proofreading and editing copy, massaging content to ensure a common voice for the site, and creating new copy.

Technical : The technical designers and programmers are responsible for server administration and the development or integration of site production tools and web site applications. They advise the other teams regarding technology-related opportunities and limitations.

Project Management : The project manager keeps the project on schedule and within budget. He or she facilitates communication between the other teams and the clients or internal stakeholders.

The success of a web site design and production project depends on successful communication and collaboration between these specialized team members. The success of a web site design and production project depends on successful communication and collaboration between these specialized team members. For the information architect, communication is a special challenge because of the intangible nature of the work. The information architect has to identify both the goals of the site and the content that it will be built on. Once you have collected the data and developed a plan, you need to present your ideas for information architecture and move the group toward consensus. All in all, this significantly burdens the architect to communicate effectively.

3.4 Organization Information

Need of organization Information: We organize

- to understand
- to explain
- to control

As information architects, we organize information so that people can find the right answers to their questions.

Organizing information involves three steps

- **Structuring:** Structuring information means determining appropriate levels of granularity for information atoms in your site and deciding how to relate them to another.
- **Grouping:** Grouping information means grouping the linked information atoms into meaningful and distinctive categories.
- **Labeling:** Labeling information means figuring out what to call these categories.

Among these structuring and grouping are considered mainly as organizing and labeling is done as a separate step.

3.5 CHALLENGES OF ORGANIZING INFORMATION

In recent years, increasing attention has been focused on the challenge of organizing information. People have struggled with the difficulties of information organization because of powerful revolution of global Internet. The Internet is forcing the responsibility for organizing information on more of us each day. How many corporate web sites exist today? How many personal home pages? What about tomorrow? As the Internet provides us all with the freedom to publish information, it quietly burdens us with the responsibility to organize that information.

High rate of growth of information: New technologies open the floodgates for exponential content growth. Now the world produces between 1 and 2 Exabyte of unique information per year. Given that an Exabyte is billon gigabytes. This growing mountain of information should keep us all busy for a while. In unorganized information space requires a lot of time to search the point where to insert the new information atoms.

1) Ambiguity: Classification systems are built upon the foundation of language, and language is often ambiguous. That is, words are capable of being understood in two or more possible ways. This ambiguity results in a shaky foundation for our classification systems. When we use words as labels for our categories, we run the risk that users will miss our meaning. This is a serious problem.

2) Heterogeneity: Heterogeneity refers to an object or collection of objects composed of unrelated or unlike parts. At the other end of the scale, homogeneous refers to something composed of similar or identical elements. This homogeneity allows for a structured classification system.

Most web sites, on the other hand, are highly heterogeneous in two respects. The heterogeneous nature of web sites makes it difficult to impose highly structured organization systems on the content.

3) Differences in Perspectives: The fact is that labeling and organization systems are intensely affected by their creators' perspectives. We see this at the corporate level with web sites organized according to internal divisions or org charts. In these web sites, we see groupings such as marketing, sales, customer support, human resources, and information systems. How does a customer visiting this web site know where to go for technical information about a product they just purchased? To design usable organization systems, we need to escape from our own mental models of content labeling and organization.

4) Internal Politics: Politics exist in every organization. Individuals and departments constantly position for power or respect. Because of the inherent power of information organization in forming understanding and opinion, the process of designing information architectures for web sites and intranets can involve a strong undercurrent of politics. The choice of organization and labeling systems can have a big impact on how users of the site perceive the company, its departments, and its products. Politics raise the complexity and difficulty of creating usable information architectures.

3.6 ORGANIZING WEB SITES AND INTRANETS

The organization of information in web sites and intranets is a major factor in

determining success, and yet many web development teams lack the understanding necessary to do the job well. Our goal in this chapter is to provide a foundation for tackling even the most challenging information organization projects.

Organization systems are composed of organization schemes and organization structures. An organization scheme defines the shared characteristics of content items and influences the logical grouping of those items. An organization structure defines the types of relationships between content items and groups.

Before diving in, it's important to understand information organization in the context of web site development. Organization is closely related to navigation, labeling, and indexing. The hierarchical organization structures of web sites often play the part of primary navigation system. The labels of categories play a significant role in defining the contents of those categories. Manual indexing is ultimately a tool for organizing content items into groups at a very detailed level. Despite these closely knit relationships, it is both possible and useful to isolate the design of organization systems, which will form the foundation for navigation and labeling systems. By focusing solely on the logical grouping of information, you avoid the distractions of implementation details and design a better web site.

3.6.1 Organization Schemes

Various organization schemes are being used today. These schemes can be divided into two categories:

1. **Exact organization schemes**
2. **Ambiguous organization schemes**

1. Exact organization schemes : These schemes divide information into well defined and mutually exclusive sections. User can search the information only if he knows what he is looking for and he knows the label so that he can identify the group / section in which the item is. This is called "known-item" searching. No ambiguity is involved.

Advantages

- Exact organization schemes are relatively easy to design and maintain because there is little intellectual work involved in assigning items to categories.
- They are also easy to use.

Disadvantages

- Exact organization schemes require the user to know the specific name of the resource they are looking for.

Examples:

1) Alphabetical : In this scheme all the information atoms are arranged alphabetically and are grouped accordingly, i.e. atoms starting with 'A' letter put together information in one group and so on. An alphabetical organization scheme is the primary organization scheme for encyclopedias and dictionaries.

2) Chronological : Certain types of information lend themselves to chronological organization. For example, an archive of press releases might be organized by the date of release. As long as there is agreement on when a particular event occurred,

chronological schemes are easy to design and use.

3) Geographical : Place is often an important characteristic of information. We travel from one place to another. We care about the news and weather that affects us in our location. Geographical organization schemes are fairly straightforward to design and use.

2. Ambiguous organization schemes: Ambiguous organization schemes divide information into categories that resist exact definition. They are mired in the ambiguity of language and organization, not to mention human subjectivity. Sometime we don't always know what we're looking for. In some cases, you simply don't know the correct label. In others, you may only have a vague information need that you can't clear. For these reasons, information seeking is often iterative and interactive and ambiguous organization schemes so useful. But Ambiguous organization scheme is difficult to design and maintain. They can be difficult to use also.

Examples:

1) Topical : Organizing information by subject or topic is one of the most challenging yet useful approaches. Few web sites should be organized solely by topic; most should provide some sort of topical access to content. In designing a topical organization scheme, it is important to define the breadth of coverage. Some schemes, such as those found in an encyclopedia, cover the entire breadth of human knowledge

2) Task-oriented : Task-oriented schemes organize content and applications into a collection of processes, functions, or tasks. These schemes are appropriate when it's possible to anticipate a limited number of high-priority tasks that users will want to perform. On today's Web, task-oriented organization schemes are less common, since most web sites are content rather than application intensive.

3) Audience-specific : In cases where there are two or more clearly definable audiences for a web site or intranet, an audience specific organization scheme may make sense. This type of scheme works best when the site is frequented by repeat visitors who can bookmark their particular section of the site. Audience-oriented schemes break a site into smaller, audience specific mini-sites.

4) Metaphor-driven : Metaphors are commonly used to help users understand the new by relating it to the familiar. Metaphors can help users understand content and function intuitively. In addition, the process of exploring possible metaphor-driven organization schemes can generate new and exciting ideas about the design, organization, and function of the web site

5) Hybrid organization schemes : The power of a pure organization scheme derives from its ability to suggest a simple mental model for users to quickly understand. This hybrid scheme includes elements of audience-specific, topical, metaphor based, and task-oriented organization schemes.

3.6.2 Organization Structures

Organization structure plays an intangible yet very important role in the design of web sites. The structure of information defines the primary ways in which users can

navigate. Major organization structures that apply to web site and intranet architectures include

- the hierarchy
- the database-oriented model
- hypertext

Each organization structure possesses unique strengths and weaknesses.

1) The hierarchy : A top-down approach

In this model we give the groups a hierarchical structure having parent child relationship between them such that they get divided into mutually exclusive groups.

Examples:

- Family trees are hierarchical.
- Organization charts are usually hierarchical
- We divide blocks into chapters into sections into paragraphs into sentences into words into letters.

Advantages

- The mutually exclusive subdivision and parent child relationship of hierarchies are simple and familiar.
- Users can easily and quickly understand web sites that use hierarchical organization models.
- They are able to develop a mental model of the site's structure and their location within that structure. This provides context that helps users feel comfortable.
- The top-down approach allows you to quickly get a handle on the scope of the web site without going through an extensive content inventory process. You can begin identifying the major content areas and exploring possible organization schemes that will provide access to that content.

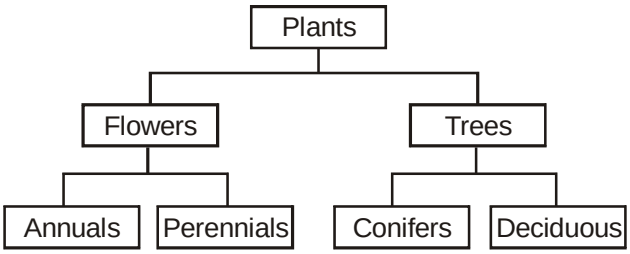


Fig. 3.4

Because hierarchies provide a simple and familiar way to organize information, they are usually a good place to start the information architecture process.

Designing the model: While designing the hierarchical model we should take care of the following points:

a) Balance between exclusivity and inclusively: The hierarchical categories should be mutually exclusive. Within a single organization scheme, you will need to balance the tension between exclusivity and inclusively. Hierarchical model that allows listing is known as polyhierarchical model. But if too many items are cross listed the hierarchy looses its value.

b) Balance between breadth and depth: It is important to consider the balance between breadth and depth in your information hierarchy. Breadth refers to the number of options at each level of the hierarchy. Depth refers to the number of levels in the hierarchy. If a hierarchy is too narrow and deep, users have to click through an inordinate number of levels to find what they are looking for. If a hierarchy is too broad and shallow, users are faced with too many options on the main menu and are unpleasantly surprised by the lack of content once they select an option.

2. The database model : A bottom-up approach. Metadata is the primary key that links Information Architecture to the design of database scheme. Metadata allows us to apply the structure and power of relation database to the heterogeneous, unstructured environments of web sites and intranets. By tagging documents and other information objects with controlled vocabulary metadata, we enable powerful searching and browsing. This is a bottom up solution that works well information large distributed environment.

Its not necessary for information architects to become experts in SQL, XML schema definitions, the creation of ER Diagrams and the design of relation database, though these are all externally valuable skills. Instead, information Architects need to understand how metadata, controlled vocabularies, and database structures can be used to enable:

- Automatic generation of alphabetical indexes.
- Dynamic representation of associative "see also" links
- Field Searching
- Advance filtering and sorting of search results.

The database model is particular useful when applied within relatively homogenous sub sites such as product catalogs and staff directories.

3. Hypertext model : Hypertext is a relatively new and highly nonlinear way of structuring information. A hypertext system involves two primary types of components: the items or chunks of information which are to be linked, and the links between those chunks. These components can form hypermedia systems that connect text, data, image, video, and audio chunks. Hypertext chunks can be connected hierarchically, non-hierarchically, or both in hypertext systems, content chunks are connected via links in a loose web of relationships.

Advantages

- This model provides you with great flexibility.
- This model allows for useful and creative relationships between items and areas in the hierarchy. If usually makes sense to the first design the information hierarchy and then identify ways information which hypertext can complement the hierarchy.

Disadvantage

- This model introduces substantial potential for complexity and user confusion because hypertext links reflect highly personal associations.
- As user navigate through highly hyper textual websites it is easy for them to get lost.
- Hyper textual links are often personal information nature. The relationship that one person sees between content items may not be apparent to others.

SUMMARY

1. The combination of organizing, labeling and navigation schemes within the information system is called Information Architecture and A person who creates the structure or map of information which allows others to find their personal paths to knowledge and who organizes the patterns inherent in data, making the complex clear is called Information Architect.

2. The information architect must communicate effectively with the web site development team. This is challenging, since information architecture is highly abstract and intangible. Besides communicating the architecture verbally, documents (such as blueprint diagrams) must be created in ways that can be understood by the rest of the team regardless of their own disciplinary backgrounds.

3. Organization structure plays an intangible yet very important role in the design of web sites. The structure of information defines the primary ways in which users can navigate. We organize information so that people can find the right answers to their questions.

4. The information architect must communicate effectively with the web site development team. This is challenging, since information architecture is highly abstract and intangible. Besides communicating the architecture verbally, documents (such as blueprint diagrams) must be created in ways that can be understood by the rest of the team regardless of their own disciplinary backgrounds.

5. Organization systems are composed of *organization schemes* and *organization structures*. An organization scheme defines the shared characteristics of content items and influences the logical grouping of those items. An organization structure defines the types of relationships between content items and groups.

6. Exact organization schemes divide information into well defined and mutually exclusive sections. User can search the information only if he knows what he is looking for and he knows the label so that he can identify the group /section in which the item is. This is called “known-item” searching. No ambiguity is involved.

7. Ambiguous organization schemes divide information into categories that resist exact definition. They are mired in the ambiguity of language and organization, not to mention human subjectivity. Sometime w*e don’t always know what we’re looking for.* In some cases, you simply don’t know the correct label.

SHORT QUESTION ANSWER

Q.1 What are the advantages & limitations of Collaboration and Communication ?

Ans. In the early days of the Web, web sites were often designed, built, and managed by a single individual through sheer force of will. This webmaster was responsible for assembling and organizing the content, designing the graphics, and hacking together any necessary CGI scripts.

Then companies began to demand more of their sites and, consequently, of their webmasters. Simple home pages quickly evolved into complex web sites. People wanted more content, better organization, greater function, and prettier graphics. Extensions, plug-ins, and languages proliferated. Tables, VRML, frames, Shockwave, Java, and ActiveX were added to the toolbox. No mortal webmaster could keep up with the rising expectations and the increasing complexity of the environment.

Increasingly, webmasters and their employers began to realize that the successful design and production of complex web sites requires an interdisciplinary team approach. An individual cannot be an expert in all facets of the process.

Rather, a team of individuals with complementary areas of expertise must work together. The composition of this team will vary, depending upon the needs of a particular project, available budget, and the availability of expertise.

However, most projects will require expertise in marketing, information architecture, graphic design, writing and editing, programming, and project management.

Q.2 Outline the major phases in website development.

Ans. There are numerous steps in the web site design and development process. From gathering initial information, to the creation of your web site, and finally to maintenance to keep your web site up to date and current. The exact process will vary slightly from designer to designer, but the basics are generally the same as:

- Information Gathering
- Planning
- Design
- Development
- Testing and Delivery
- Maintenance

Q.3 Describe the role of information architect in detail.

Ans. The main jobs of the information architect are given below. An information Architect:

- Clarifies the mission and vision for the site, balancing the needs of its sponsoring organization and the needs of its audiences.
- Determines what content and functionality the site will contain.
- Specifies how users will find information in the site by defining its organization, navigation, labeling, and searching systems.
- Maps out how the site will accommodate change and growth over time.

Q.4 What is Exact organization and Ambiguous organization Schema?

Answer: Exact organization schemes: These schemes divide information into well defined and mutually exclusive sections. User can search the information only if he knows what he is looking for and he knows the label so that he can identify the group /section in which the item is. This is called "known-item" searching. No ambiguity is involved.

Ambiguous organization schemes: Ambiguous organization schemes divide information into categories that resist exact definition. They are mired in the ambiguity of language and organization, not to mention human subjectivity. Sometime *we don't always know what we're looking for.* In some cases, you simply don't know the correct label. In others, you may only have a vague information need that you can't clear. For these reasons, information seeking is often iterative and interactive and ambiguous organization schemes so useful. But Ambiguous organization scheme is difficult to design and maintain. They can be difficult to use also.

Q.5 What are different steps involve in organizing information?

Answer: It involves following steps:

i. **Structuring**: Structuring information means determining appropriate levels of granularity for information atoms in your site and deciding how to relate them to another.

ii. **Grouping**: Grouping information means grouping the linked information atoms into meaningful and distinctive categories.

iii. **Labeling**: Labeling information means figuring out what to call these categories.

iv. **Finding and managing**: Finding is need of users. Managing is goal of business.

v. **Art and Science**: Information Architecture is both art and science. Information Architecture must rely on experience, intuition and creativity, willing to take risks. This is the art of information Architecture. Information Architecture must follow the rules for organizing the information space. This is the science of information Architecture.

IMPORTANT QUESTIONS

Q.1 What is Information Architecture? What are its different components? Explain each in detail.
Q.2 Describe the role of Information Architect in detail.
Q.3 What are the various challenges of organizing information?
Q.4 How websites are organized? Explain.
Q.5 What are different organizational schemes used by information architect.
Q.6 What are the advantages & limitations of Collaboration and Communication?
Q.7 What are different approaches used in organization structure?

****

CREATING COHESIVE WEBSITES

IN THIS CHAPTER, YOU WILL LEARN

- Creating Cohesive Organization Systems
- Conceptual Overview for Website Design
- Website Design Issues
- Navigation System
- Integrated Navigation Elements
- Designing Elegant Navigation Systems
- Searching Systems
- Searching Your Website
- Designing the Search Interface
- Conceptual Design
- Indexing The Right Stuff
- Grouping Content
- Architectural Page Mockups
- Design Sketches

4.1 CREATING COHESIVE ORGANIZATION SYSTEMS

Organization systems are fairly complex. We have so many options for choosing appropriate organization scheme and appropriate organization structure. Taken together in the context of a large website development project, the choice of a proper system, becomes difficult, that's why it is important to break down the site into its components, so you can take one option for scheme/structure at a time. Also we know that all information retrieval system work best when applied to narrow domains of homogenous content, we can identify opportunities for highly effective organization system. However it's also important not to loose sight of the big picture.

In considering which organization schemes to use, remember the distinction between exact and ambiguous schemes. Exact schemes are best for known-item searching. Ambiguous schemes are best for browsing and associative learning. Whenever possible use both types of schemes. Also beware of the challenges of organizing information. When thinking about which organization structures to use, keep in mind that large web sites and intranets typically require all three types of structure. The top-level, umbrella architecture for the site will almost certainly be hierarchical. As you are designing this hierarchy, keep a lookout for collections of structured, homogeneous information. These potential subsites are excellent candidates for the database model. Finally, remember that less structured, creative relationships between content items can be handled through hypertext. In this way, all three organization structures together can create a cohesive organization system.

4.2 CONCEPTUAL OVERVIEW FOR WEBSITE DESIGN

Websites can have many purposes and the aim of your website, i.e. what you wish to achieve with your website, is important to establish before any work is begun. A website is a tool. A tool that potentially has many functions. It is the aim of the website that defines its function.

- Your website may be an informational website, aiming to educate and stimulate, and an online community where people share ideas and thoughts in forums or chat rooms. You may be thinking about a website for your business to increase cost efficiency and increase your client base.

- Your website may have a particular functional aspect such as allowing people to order products securely online. This has the benefit of creating customer convenience and repeat orders and your business can reap the benefit of lower overheads with more competitive product pricing.

- A big plus is that it allows your company or service to work more efficiently and effectively by improving communications. This could be a private area of your public website with staff only access or a dedicated site for your staff.

- Your website may have global appeal or it may focus on a particular geographical area. It may be aimed a particular niche or section of society or it may be of wider interest. Who the website is aimed at - your audience, is crucial.

- Your website may be public or may have restricted access. The content may be free to anyone or people may use their credit card to pay for a subscription to your sites content. It may be commercial or non-profit. You may wish to have sponsors who advertise on the site in the form of banners or pop-up “web-verts”.

- You may simply wish to have an online brochure giving people basic information about your company, from which you can expand and build your web presence, paving the way for future developments.
- The purpose and aim of your website is only limited by your imagination.

There are three main elements to websites: the organization of the content and how it is navigated, the appearance of the web pages and the content in its own right. This is very much a pre-production planning stage which is very important if we are to implement a website effectively within a timescale & budget. If you want your website tying in with any existing promotions, colour schemes, or decor, let us know as we can integrate your website into your existing image. If you have a logo or any graphics you use, or wish to use, let us know, so we can incorporate them to create a branding for your website. We can also create you a logo should you need one.

Web Pages can also be interactive, that is they respond to a user's actions. This can be simple, such as the colour of a link changing colour when the user places their cursor over it, to a fully animated button with personality.

The third element of a webpage, the content, is up to you. We can incorporate most kinds of digital media into a website or digitize analogue information for you, for inclusion in the website, e.g. scanning of photos & recording of audio or video. This includes written text, video, audio, music, speech, pictures, animation, diagrams, photos, PDF documents, Microsoft Office documents, CAD files, anything you can think of really.

We can capture your content if you do not have the facilities such as video shooting and editing, taking photos, scanning hardcopy drawings/diagrams, recording audio and encoding. We can also create interactive animations and product presentations. Please see our other services section for further details.

We will also optimize your media for the Internet so that it has a smaller file size and therefore downloads faster. This applies to video, pictures and sound.

As a word of warning the larger the file size of your content, the longer it will take to download when someone is browsing your website. Most people have dial-up internet connections which run at a maximum of 56kbps and realistically this is not a fast enough connection to stream video of significant quality, even when optimised. However sound, pictures and animation are fine if optimised correctly. If wish to use extensive video streaming it is better to aim at broadband users.

4.2.1 Defining your site's content areas

Defining your content areas will help you to develop your navigational

structure. This step is best done in a group of three to five people so that each person can represent the profiles from one or two of your user data sheets.

First, analyze the content you already have, either in print or on the Web and decide which pieces should be added to your new site, updated or discarded. Keep or update only the content that will be useful to your users.

Next, list all of the content areas that your users will want to find on your site. The ideal way to do this is to ask a wide sampling of actual users (who are members of your target audiences) what they will be seeking ("Conducting user interviews and creating data sheets"). If you do not have access to actual users, you will need to imagine what they will want to find on your site. Once you've done this, you may need to set aside any user goals that are not practical to include in the scope of your project. Also, you may need to add items your key stakeholders want to include.

The "Content area brainstorming" activity on the next page is an exercise that can help you to further define the content areas of your site.

4.2.2 Accessible Web Design

In the context of web site design, accessibility is a measure of how easy it is to access, read, and understand the content of a web site. Accessibility is complicated by the fact that a web site is not a published piece of work so much as a living document that can be interpreted in different ways by different browsers and on different platforms. Web sites are not a print medium- although they are most often read in a visual manner, there are many different ways a web page can be experienced, such as via a speech browser or via an indexing robot. A web page is a combination of textual information which is interpreted appropriately by a browser and linked to files of various types, such as graphics, movie clips and sound files.

Since a web page can be interpreted differently by different browsers with different capabilities, and since the language of a web page- HTML, is constantly evolving, accessibility must be considered to make a page usable by as many people as possible. The keys to making your page accessible are graceful degradation, standards compliance, fast loading, and intelligent organization.

Graceful Degradation : Since HTML is continually changing and different browsers support different elements, graceful degradation is the key to making sure that pages are readable and accessible in all browsers. When a browser encounters tags it doesn't understand or can't display, degradation takes place. Whether this degradation will cause some of your page content to be lost to the browser, or whether the content of your page can still be accessed fully is dependent on whether the degradation is graceful.

The HTML standards were written with graceful degradation in mind- new attributes to older tags are safely ignored so that the rest of the tag can still function normally, and new tags are written with alternative display for browsers that don't support them in mind. There are many elements of HTML that can't be displayed or can be turned off in browsers that were written with the knowledge of these elements- such as images, java, and frames. Using the appropriate methods to provide an alternative message to those who can't see those elements or have

turned them off is one way to design for graceful degradation.

If you design pages with graceful degradation in mind, by utilizing the built in elements of the HTML standards, and the advice provided here, you can design pages that should degrade gracefully in all browsers and are accessible.

4.2.3 Website Planning

It is the first most important part of building a website. Before designing a website, it is necessary to do proper planning as in the case of building a house. If you start building your house without giving a thought of planning, the house may end up into an uncomfortable dwelling. Same way, for building a good website certain things should be taken care of. The following "Things to consider" while planning a website:

- Purpose of website
- Website contents
- Directory structure of the website
- Target audience
- Blue print of a website
- Budgeting

1. Purpose of website: The initial stage of planning your website is to "Identity the Purpose of the Site". It could be as follows:

- To educate people
- To entertain people
- To promote / sell a product
- To promote information on companies, products and sources, sports and games, Travel and tourism, medicines and health.

Having specific purpose is a necessary since it provides you with the guidelines on what should be put onto your site; what resources (Software, Hardware, and Manpower) are needs for the project etc.

2. Target Audience: The next step is to "Identify the Target Audience", their knowledge, background, interests and needs, age, gender, geographic location etc. Determining, who the visitors are likely to be crucial in deciding not only the general appearance (look and feel) of the site, but also the technology that might be used to build the site.

3. Website contents: Once the Purpose and Target Audience of the website are defined properly, organize the website contents. Put together any existing documents and pictures you want to work. For example, if it is a company website, you may want to assemble logos, company information, and product descriptions.

4. Blue print of a website: Before designing your website, you may like to create

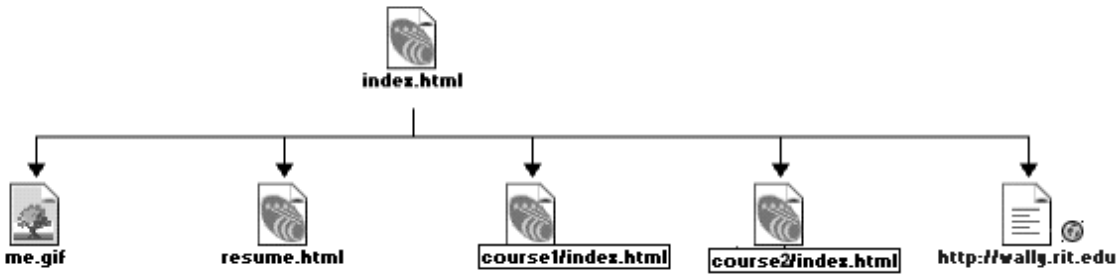


Fig. 4.1

This will give you complete Hierarchical Picture of the links among different pages in the web site.

5. Directory Structure of the website: Organizing your site carefully from the start can save your frustration and time later on. For better management of the site, it would be better to create a directory structure for your site, where the website contains all the files put under one directory. This may hold some sub-directories containing related items like you can have subdirectories of images or pictures, sound clippings. So breakdown your site into different categories and store concerned files inside. Developing a logical structure is an essential part of developing your website.

6. Budgeting: For a large and complex web site, it is also very important to do budgeting and other costs involved for the development of a web site. While doing the budgeting the following things need to consider:

- Salaries and benefits of development and support staff
- Hardware and Software used
- Server and technical support

Other costs involved:

- Domain name and registration
- Internet Service Provider
- Search Engine Registration

4.3 WEBSITE DESIGN ISSUES

Sites with different objectives will obviously have different needs. Moreover, individuality and uniqueness of web sites are also valued features. With that in mind, those guidelines are offered as a starting point for developing good web design skills, not as formula that should be followed point by point.

a) Page Loading Efficiency

- The temptation to overload a page with graphics should be resisted.
- A few well-chosen graphics are fine, but too much on a page and the visitor may become frustrated with the required time to load a page, and "click, click" they are off to another site.
- Frames also increase the loading time, and if the site sells or exchanges advertising space in which banners will appear, these items will also slow down the load time.

b) Simplicity

- Avoid clutter on web pages.
- If the business has a lot of information to convey, organize it well and spread it out over multiple pages.
- A guideline is to use about 60 characters per line.

- Also, avoid long pages that required a lot of scrolling.
- Again, organizing the material well can preclude excessive scrolling from being necessary.

c) Use the Space Wisely

- Do not ramble on; make each statement count.
- Kernels of information that are succinctly worded and have impact are best.

d) Create a reason to return

- Once a visitor comes to the site, give them a reason to return.
- Suggest they bookmark the site-it works!

e) Framing

- A frame is a section of the viewer's computer screen.
- A screen can be split into multiple sections that can load different web pages, even those from other sites.
- The use of frames has its benefits and its drawbacks.

f) Tables and Fonts

- Table are useful for providing structure to text that will not be lost due to the size of the visitor's screen and the size of viewing window, which is affected by viewer's web browser.
- Whenever possible, avoid using all uppercase letters as they are more difficult for the eye to follow.
- The TIMES and HELVETICA fronts for readability on websites.

g) Graphics

- Graphics can enhance a web site when used properly.
- Attempt to use images that are no larger than 70k or the load time may annoy visitors.
- Images that gradually appear sharper are called interlaced graphics.
- Not everyone appreciates these pictures, and some people find them annoying.

h) Purchasing Information

- Sites that sell their products/services on-line should clearly post policies in an easily found place regarding these items:
- Tax rates;
- Shipping rates;
- Shipping schedules;
- Return policy;
- Privacy of transaction; and
- Security of data that is transmitted.

i) Tracking Data

- In order to analyze the success or contribution of site, certain data need to be tracked.

Some useful information includes:

- Number of different visitors(not repeat visitors);
- Number and frequency of repeat visitors;
- Location of site prior to visit, including the search engine used to locate the site, if applicable;
- Length of time of visit;
- Pages visited
- Items examined by visitors;
- Domain names of visitors;
- Country codes of visitors; and
- Purchase made, if applicable.

4.4 Navigation System

When we have lards amount of information then to organize the information space we divided the information space into groups and label them. To look for any information atom we need to search for its link information its group. This is done using browsing/ navigation user can get lost in the information space but a well-designed taxonomy may reduce the chances that user will become lost. So generally Navigation Systems are beneficial if information is organized using the hierarchy model. Complementary navigation tools are often needed to provide context and to allow for greater flexibility.

Navigation has two primary Elements:

- Location Indicator
- Navigation Controls

4.4.1 Location Indicator

Location indicators let users know where they are in the site at the moment.

It needs to be done in mind that users coming from outside can enter at any page, not necessary in a 'main' page. They need to be able to orientate themselves. Equally it is important that users navigate around the web site have a clear idea of where they are both in absolute terms and in relation to other content. There are a number of ways of presenting information, which you choose really depends on the complexity of the content and the depth of the site structure. Whatever method is chosen there are a few essentials in good location system.

- **Consistency:** Location information should appear on EVERY page of the site, in same place and in the same style.
- **Completeness :** Location indicator should tell the user precisely where they are and should be clear even to a user who has entered the site at internal page.
- **Clarity:** The location indicator should be identifiable for what it is and male sense in the context of other navigation.

How to Indicate Location

- In simple site, naming the page will be sufficient. For this to work the page should also appear in the main navigation so that it is relevant within the overall structure of the site.
- Page banners can also work with subsidiary navigation that is the banner has the same name as an item that appears in navigation specific to that section
- A 'crumb tai;' can be used.

  e.g. Home > School Information > Departments > Sport > Tennis
- Color can be used. For example a different background color, contrast color or sidebar in each part of the site. To be really effective the color change should be reflected in the navigation.
- Graphics can be used, although it is quite difficult to do this successfully.

4.4.2 Navigation Controls

Navigation controls are the main navigation links and allow users to move around the site. Whether they comprises of images and text, they should be predictably located in the same place, and within same appearance, on each page.

These have several purposes:

- To allow users to move about within the site
- To tell user what information is available at the site
- To work with location indicator to orientate users

Navigation controls can be divided into 3 categories:

1) Main navigation : should appear on all pages in the same style and in the same place. Choosing to place it on the left or the top is generally a good idea since this is where people expect to find it. Having broken your content into categories you need to link to each category in the main navigation. Some categories will be single page (e.g. Home page) others will be whole directories of pages (e.g. School Information). It helps to think of each section with a lot of contents as mini sites, each with its own home page to which the main navigation link goes and which can be used to further orientate the users. Each can also have own subsidiary navigation if it contains multiple pages. Link to other sites should never appear in the main navigation controls.

2) Subsidiary Navigation: Navigation specific to the section of the site page is in. Having subsidiary menus that only appears once a section, or mini sites is entered, can successfully combined navigation and location indicator. One approach is to have a sub menu drop down from main menu item once that section is entered. Another approach is to separate the section's navigation from the main navigation, perhaps having the main navigation at the top of the page and section specific navigation at the left, or the main navigation on the left and the section navigation on the right. Whatever method is chosen it should be used consistently throughout the site.

3) Secondary Navigation: Secondary Navigation is important for at least three reasons:

- If the user lost, then a drop down box, search function or site map provides an opportunity for them to quickly reoriented them or get back to a familiar page.
- A site map page can provide more detail about what is in each section of the site than a navigation bar can.
- Secondary navigation provides a fast way for user to get content at a deep level, especially in areas of the site where there is a lot of content.

Some general consideration about Navigation:

a) Keep in Short: The main navigation should keep as short and simple as possible. If you find that you need more than about 10-12 links in your main navigation, you may need to rethink your site organization.

b) Provide alternatives: of there are a lot of information at the site, a prominent link to a site map where a more descriptive content page is available should be included in main navigation. Use of drop down box or search function can also provide useful alternative routes to content.

c) Divide and rule: In large site it is sometimes useful to add a quick means of reaching a few important pages. Thus you will often see something like this at the top of the page: Home I Contact Us I site map or split navigation with a separate navigation bar for six or seven main pages or section in the site. Usually this sort of navigation bar is located at the top of the page.

d) Take me Home! : The main site logo should always link to home page. If a different logo or graphic is used to indicate sections these should link to the main section page. Users expect this and come to rely on it.

e) Provide a text version: Whatever style of navigation you choose, include a text version at the bottom of each page. This is not an invariable rule, some site design may make it superfluous, but surer do generally expect it.

4.5 TYPES OF NAVIGATION SYSTEMS

A complex web site often includes several types of navigation systems. To design a successful site, it is essential to understand the types of systems and how they work together to provide flexibility and context.

A) Embedded / integrated Navigation Systems: Embedded Navigation Systems are typically wrapped around and infused within the content of the site. These systems provide context and flexibility helping users understand where they are and where they can go. Embedded Navigation Systems can be further divided into three categories :

1. Global (site-wide) Navigation System: By definition, a global navigation system is intended to be present on every page throughout a: site. It is often implementing in the form of a navigational bar

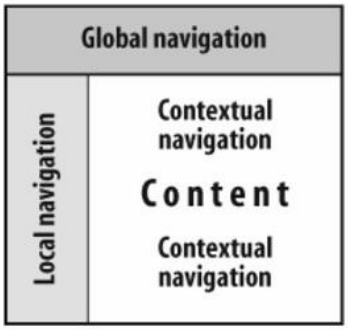


Fig. 4.2 : Global, local, and contextual embedded navigation systems.

at the top of each page. These site wide navigation systems allow direct access to key areas and functions, no matter where the user travels in the site's hierarchy. Most global navigation systems provide a link to the home page. Many provide a link to the search function.

2. Local Navigation Systems: Local Navigation Systems enable users to explore the immediate area. Some lightly controlled sites integrate global and local navigation into a coexistent unified system. A user who selects business sees different nevi anion options than a reader who selects sports, but both sets of options area presented within the same navigation framework. These local navigation systems and the content to which they provide access are often so different that these local areas are referred to as sub sites or sites within sites. Sub sites exist because (1) areas of content and functionality really do merit a unique navigation approach (2) due to decentralized nature of large organization different groups of people are often responsible for different content areas and each group may decide to handle navigation differently.

3. Contextual Navigation system: Some relationships don't fit neatly into the structured categories of global and local navigation. This demands the creation of contextual navigation links specific to a particular page, document or object. E.g. Words or phrases within sentences are represented as embedded or inline hypertext links. On an e-commerce site, these "See Also" links can point users to related products and services. In this way contextual navigation supports associative learning. Users learn by exploring the relationship you define between items. They might learn about useful products they didn't know about. Contextual Navigation system also called as AdHoc Navigation System.

B) Supplemental/ Remote Navigation System: These navigation systems are external to the basic hierarchy of a website and provide complementary ways of finding content and completing tasks. These navigation systems provide users with and emergency backup. Some of the examples of Remote navigation Systems are:

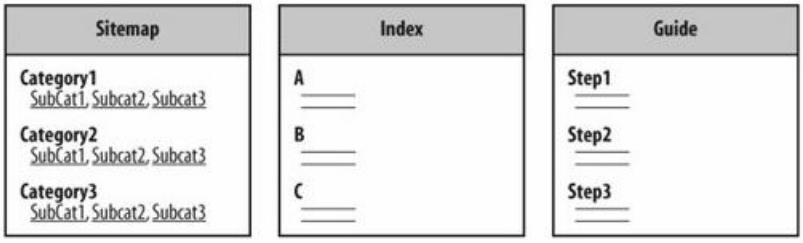


Fig. 4.3 Supplemental Navigation System.

Supplemental navigation systems :

1) Sitemaps: In a book/magazine, the table of contents presents the top few levels of the information hierarchy. It shows the organization structure for the printed work and supports random as well as linear access to the content through the use of chapter and page numbers. In context of websites a sitemap provides a board view of the content in the website and facilities random access to segmented portions of that content. A sitemap can employ graphical or text based links to provide the user with direct access to pages of the site. A sitemap is the most

natural for websites that lend themselves to hierarchical organization. But for a small website with only two or three hierarchical levels a sitemap may be unnecessary.

2) Site Indexes : Similar to the back of book index found in many print materials, a web based index presents keywords organization phrases alphabetically, without representing the hierarchy. Unlike a table of contents indexes are relatively flat, presenting only one or two levels of depth. Therefore indexes work well for users who already know the name of the item they are looking for. Large complex Websites often require both a sideman and a site index. For small sites, a site index alone may be sufficient. A major challenge in indexing a website involves the level of granularity.

Methods to create index are :

a) For small sites create content to inform decisions about which links to include.

b) For large sites, use controlled vocabulary indexing at the document level to drive automatic generation of tie site index.

3) Guides : Guides take several forms including guided tours, tutorials and micro portal focused around a specific audience topic or task. In each case, guides supplement the existing means of navigating and understanding site content. Guides typically feature linear navigation but hyper textual navigation should be available to provide additional flexibility.

Uses of Guides :

1. Guides often serve is a useful toil for introducing new users to the content and functionality of a website.
2. Guides can be valuable marketing tools for restricted access websites enabling you to show potential customers what they will get for their moneys.
3. Guides can be valuable internally, providing an opportunity to showcase key features of a redesigned site to colleagues, managers and venture capitalists.

Note: Linking between Navigation and searching: Searching is loosely linked with integrated Navigation Systems and tightly linked with Remote Navigation systems.

4.6 INTEGRATED NAVIGATION ELEMENTS

In global and local navigation systems, the most common and important navigation elements are those that are integrated into the content-bearing pages of the web site. As users move through the site or sub-site, these are the elements they see and use again and again. Most integrated navigation elements fit into one of two categories: navigation bars and pull-down menus.

Navigation Bars : A navigation bar is a collection of hypertext links grouped together on a page. Alternatively, the navigation bar may be graphical in nature, implemented as an image map or as graphic images within a table structure. The decision to use text versus graphic navigation bars falls primarily within the realms of graphic design and technical performance rather than information architecture. Graphic navigation bars tend to look nicer but can significantly slow down the page loading speed (although, if you're able to reuse the same global navigation bar throughout the site, loading speed will only be hurt once, since the image will be

cached locally). If you do use graphic navigation bars, you need to be sensitive to the needs of users with low bandwidth connection. You should also consider those users with text-only browsers (there are still quite a few out there) and those users with high-end browsers who turn off the graphical capabilities to get around more quickly. Appropriate uses of the <ALT> attribute to define replacement text for the image will ensure that your site supports navigation for these users.

Fig. 4.4 : Global navigation bars from Dell, Apple, and Amazon

This navigation bar, which appears at the bottom of the page, demonstrates an interesting blend of graphic icons (with labels) and textual options. The global navigation icons provide a splash of color, while their labels ensure usability. The textual local navigation options allow for the creation of many footer navigation bars without restrictive costs.

1.5.3.2 Frames

Frames present an additional factor to consider in the application of textual or graphical navigation bars. Frames allow you to define one or more independently scrollable "panes" within a single browser window. Hypertextual links within one pane can control the content displayed in other panes within that same window. This enables the designer to create a static or independently scrolling navigation bar that appears on every page in that area of the web site. This frame-based navigation bar will be visible to the user in the same location in the browser window even while scrolling through long documents. By separating the navigation system from content in this way, frames can provide added context and consistency as users navigate a web site. However, frames present several serious problems, both from the consumer's and producer's perspective. Architects should proceed very carefully in considering frame-based navigation solutions.

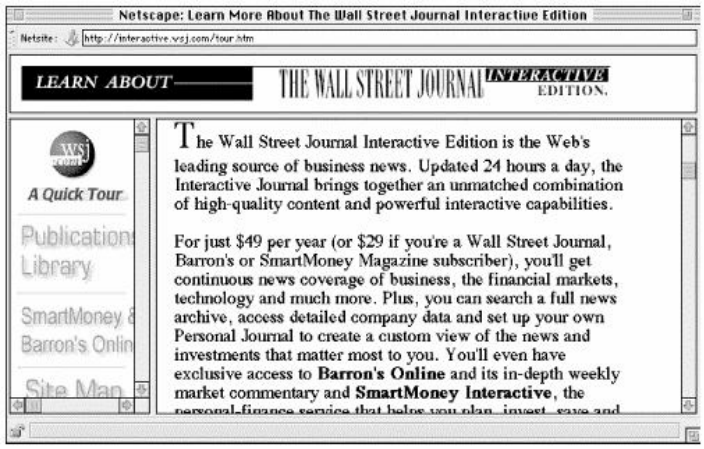


Fig. 4.5 : Frames.

The Wall Street Journal's Interactive Edition makes use of frames. It's a relatively elegant implementation, but it limits screen real estate and disables basic navigation features.

Pull-Down Menus : Pull-down menus compactly provide for many navigation options. The user can expand what appears as a single-line menu to present dozens of options (as shown in Figure 1.5.4.3). The most common pull-down menus on the Web are implemented using the standard interactive forms syntax. Users must choose an option from the menu and then hit a Go or Submit button to move to that destination.

Fig. 4.6 Pull-down Menu

This pull-down menu enables users to select a location without first going to a separate web page. This approach avoids further cluttering the main page with a long list of locations.

You can implement a more sophisticated version of the pull-down menu (also known as the pop-up menu) on the Web by using a programming language such as Java or JavaScript. As the user moves the cursor over a word or area on the page, a menu pops up. The user can directly select an option from that menu. Use pull-down and pop-up menus with caution. These menus allow designers to pack lots of options on one page. This is usually what you are working hard to avoid. Additionally, menus hide their options and force the user to act before being able to see those options. However, when you have a very straightforward, exact organization scheme, these menus can work well.

4.7 DESIGNING ELEGANT NAVIGATION SYSTEMS

Designing navigation systems that work well is challenging. You've got so many possible solutions to consider and lots of sexy technologies such as pop-up menus and dynastic site maps can distract you from what's really important: building context, improving flexibility, and helping the user to find the information they need. No single combination of navigation elements works for all web sites. One size does not fit all. Rather, you need to consider the specific goals, audience, and content for the project at hand, if you are to design the optimal solution.

However there is a process that should guide you through the challenged of navigation system design. It begins with the hierarchy. As the primary navigation System, the hierarchy influences all other decisions. The choice of major categories at the highest levels of the website will determine design of the global navigation system. Based on the hierarchy, you will be able select key pages or types of pages that should be accessible from every other page on the web site in turn, the global navigation system will determine design of the local and then ad hoc navigation systems. At each level of granularity. You design of the higher order navigation system will influence decision at the next level.

Once you have designed the integrated navigation system, you can consider the addition of on or more remote navigation elements. In most cases, you will need to choose between a table of contents, an index, and a sitemap. Is the hierarchy strong and clear? Then perhaps a table of contents makes sense. Does the hierarchy get in the way? Then you might consider an index. Does the information lend it self to visualization? If so, a sitemap may be appropriate. Is there a need to help new or prospective users to understand what they can do with site? Then you might add a guided tour.

If the site is large and complex, you can employ two or more of these elements. A table of contents and an index can serve different users with varying needs. However, you must consider the potential user confusion caused by multiple options and the additional overhead required to design and maintain these navigation elements. As always, it's a delicate balancing act. If life on the high wire unnerves you be sure to build some usability testing into the navigation system design process. Only by learning from users can you design and reline an elegant navigation system that really works.

4.8 SEARCHING SYSTEMS

Navigation and Searching both are used for finding information. Navigation searches for the information to be found by moving between links available. But information searching we give the information about the information to be found as text to the search engine and search engine does the task of finding information for users. We can search for a phrase but can't navigate.

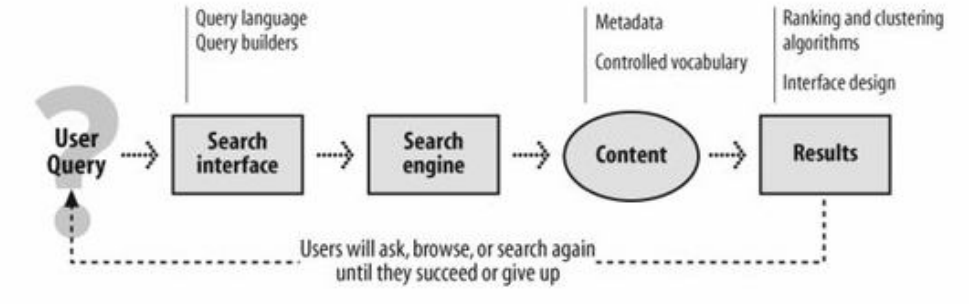


Fig. 4.7 Basic Anatomy of a Search System.

The basic anatomy of a search system

Need of Searching systems :

1. As the amount of information on the website increases it become difficult to find the required information. If the navigation systems are not properly designed and maintained then to find the required information searching systems are required.
2. If your site has enough contents and users come to your site to look for information then site need searching systems.
3. Search system should be there on your site if it contains highly dynamic contents e.g. web based newspaper.
4. A search system could help by automatically indexing the contents of a site once or many times per day. Automating this process ensures that users have quality access to your website's contents.

4.9 SEARCHING YOUR WEBSITE

Assuming you have decided to implement a Searching system for your website. It's important to understand how users really search before designing it.

Users have different kinds of information need: Some user want just a little while other wants detailed assessment of everything there is to know about .the topic. Some want only the accurate, highest quality information; while others do not care much about the reliability of source. Some will wait for results while others need the information yesterday. Some are just plan happy to get any information at all, regardless of how much relevant stuff are really missing. Users needs and expectation vary widely and so the information systems that them must recognize, distinguish and accommodate these different needs.

To illustrate let's look at one of these factors in greater detail: The variability information users searching expectations.

- **Known item searching :** Some users information needs are clearly defined and have a single correct answer. When you check the newspaper to see how your stock information amalgamated shoelace and aglet is 'doing (especially since the hostile Microsoft takeover attempts), you know exactly what you want that the information exists and where it can be found.
- **Existence searching :** However some users know what they want but do not

know how to describe it or weather the answer exists at all e.g., you must want to buy shares information Moldovan high start-ups and that carries no load. You are convinced that this sector is up and coming, but do fidelity and Merrill lynch know this as well'. You might check their Webster, call a broker or two, or ask your in the know aunt. Rather then a clear question for which a right answer exists, you have an abstract idea on concept, and you don't know whether matching information exists. The success of yours search depends as much upon the abilities of the brokers, the websites, and your aunt to understand your idea and its contexts as whether the information (information in this case a particular mutual fund) exists.

- **Exploratory searching :** Some users know how to phrase their question, but don't know exactly what they are hoping to find and are really just exploring and trying to learn more. If you ever considered changing careers you know what we mean you are not sure that you definitely what to switch to chinchilla farming, but you have heard it is the place to be, so you might informally ask a friend of a friend who an uncle in the business. Or you call the public library to see if there's a book on the subject, or you write to the chinchilla professionals association requesting more information. In any case, you are not sure exactly what you will uncover, but you are re willing to take the time to learn more. Like existence searching, you have so much a question seeking answer as much as an idea that you want to learn more about.

- **Comprehensive Searching (Research) :** Some users want everything available on a given topic. Scientific researchers, patent lawyers, doctoral students trying to find unique and original dissertation topics, and fans of any sort fit in to this category. There are many other ways of classifying information needs. But the important thing remember is that not all users are looking for the same thing. Ideally, you should anticipate the most common types of needs that your site's users will have and ensure that these needs are met minimally; you should give some thoughts to the variations and try to design a search interface that is flexible in responding to them.

4.10 DESIGNING THE SEARCH INTERFACE

Concept of Searching System : There are two models of searching systems:

1. In the first and older model user express their information need as query that they enter in a search interface. They may do so using a specialized search language.
2. In the second model users express the information need information the natural language like English.

After this step Queries are matched against an index that represent the site's content and a set of matching documents is identified.

Designing the Search Interface : With so much variation among users to account for, there can be no single ideal search interface. Following factors affect choice of search interface:

1. The levels of searching expertise users have: Are they comfortable with Boolean operators. Or do they prefer natural language? Do they need simple or high powered interface? What about a help page?

2. The kind of information the user wants: Do they want just a taste or are they doing comprehensive research? Should the results be brief, or should they provide extensive detail for each document?

3. The type of information being searched is it made up of structured fields or full texts? Is it navigation pages, destination pages, or both? HTML or other formats?

4. How much information is being searched will users be overwhelmed by the number of documents retrieved?

Support Different Modes fo Searching : Use the same interface to allow users to search the product catalog, or the staff directory, or other content areas. Are non-English speakers important to your site? Then provide them with search interfaces in their native languages. Including language specific directions, search commands and operators, and help information. Does your site need to satisfy users with different levels of sophistication with online searching? Then consider making available both a basic search interface and an advanced one.

I. Simple / Basic search interface : A simple search interface was required; because at limes users wouldn't need all the firepower of an advanced search interface. Especially when conducting simple known item searches. A simple search box is ideal for the novice or for a user with a pretty good sense of what he or she is looking for. Mammal filtering options are provided including searching for keywords within little and abstract fields, searching within the author field or searching within the publication number field. These filtering options provide the user with more power by allowing more specific searching. But because the labels keyword, Author, And publication Number are fairly self explanatory. They don't force the user to think too much about these options.

II. Advanced search Interface : We needed interface that would accommodate this important expert audience who were used to complex Boolean and proximity operators and who where already very used to the arcane search languages of other commercial information services. This interface supports the following types of searching:

- **Fielded Searching** : Author, keyword, Title, Subject and ten other fields are reachable. A researcher could, for example find a dissertation related to his or her area of interest by searching the subject field, and learn who that doctoral student's advisor was by reading the abstract. To find other related dissertations, the researcher could then search the advisor field to learn about other doctoral students who shared the same advisor.

- **Familiar Query Language :** Because many different query language conventions are supported by traditional on line products, users may be used to an established convention. The effort to support these users is made by allowing variant terms. For the field Degree Date the user can enter either ''ddt'', ''da'', ''date'', ''Yr '' or year.

- **Longer Queries :** More complex queries often require more space than the single line entry box found in the simple search interface. The more complex interface supports a much longer query.

- **Reusable Result Sets** : Many traditional online information products allow

searchers to build sets of results that can be reused. In this example, we've ANDed together the two sets that we've already found and could in turn combine this result with other sets during the iterative process of searching. Because this advanced interface supports so many different types of searching we provided a substantial help page to assist users. For users of common browsers, the help page is launched if a separate browser window so that users don't need to exit the search interface to get help.

4.11 Conceptual Design

4.11.1 What Do You Mean by Blueprint?

Blueprints are the architect's tool of choice for performing the transformation for chaos in to order. Blueprints show the relationship between pages and other content components and can be used to portray organization, navigation and labeling systems. They are often referred to as sitemaps and do in fact have much information common with those supplemental navigation systems. Both the diagram and the navigation system display the shape of the information space information overview, functioning as a condensed map for site developers and users, respectively

4.11.2 High-level Architecture blueprints

High level architecture blueprints are often created by information architects as pat of a top down information architecture process. The very act shaping ideas in to the more structure of a blueprint forces you to become realistic and practical. During the design phase, high level blueprints are most useful for exploring primary organization schemes and approaches. High level blueprints map out the organization and labeling of major areas. Usually beginning with a bird's eye view from the main page of the website.

This high-level blueprint shows pages, components within pages, groups of pages, and relationships between pages. The grouping of pages can inform page layout. For example, the three value-added guides should be presented together, whereas Search & Browse, Feedback, and News should be presented separately.

4.11.3 Creating High-Level Architecture Blueprints

These blueprints can be created by hand, but diagramming software such as Visio or OmniGraffle are preferred. These tools not only help to quickly layout the architecture Blue prints, but can also help with site implementation and administration.

Some Important points:

1. Blueprints focus on major areas and structure of site ignoring many navigation details and page level details.
2. Blueprints are excellent tools for explaining your architectural approaches.
3. Presenting blueprints information person allows you to immediately answer the questions and address client concerns as well as to explore new ideas while they are fresh in your mind and the client's.

4. As you create blueprint it is important to avoid getting locked into a particular type of layout.
5. If a meeting isn't possible, you can accompany blueprints with descriptive test based documents that anticipate and answer the most likely documents.

4.11.4 Keeping Blueprints Simple

As a project moves from strategy to design to implementation, blueprints become more utilitarian. They need be produced and modified quickly and often draw input front increasing number of perspectives, ranging from visual designers to editors to programmers. Those team members need to be able to understand the architecture. So it's important to develop a simple condensed vocabulary of objects that can explain in a brief legend.

4.12 INDEXING THE RIGHT STUFF

Searching only works well when the stuff that's being searched is the same as the stuff that users want. This means you may not want to index the entire site. We will explain:

4.12.1 Indexing The Entire Site

Search engines are frequently used to index an entire site without regarded for the content and how it might vary. Every word of every page, whether it contain real content or help information, advertisement, navigation, menus and so on. However, searching barks much batter when the information space is defined narrowly and contains homogeneous contents. By doing so, the site's architects are ignoring two very important things: that the information in their site isn't all the same. And that it makes good sense to respect the lines already drawn between different types of content. For example, it's cleared that German and English content are vastly different and that there audience's overlap very little (if at all) so why not create separately searchable indices along those divisions?

4.12.2 Search Zone: Selectively Indexing the Right Content

Search zone are subset of website that have been indexed separately from the rest of the site contents. When you search a search zone, you have through interaction with the site already identified yourself as a member of a particular audience or as someone searching for a particular type of information. The search zones in a site match those specific needs and results are improved retrieval performance. The user is simply less likely to retrieve irrelevant information. Also note the full site search option: sometimes it does make sense to maintain an index of the entire site, especially for user who are unsure where to look, who are doing a comprehensive leave no stones unturned search, or who just haven't had any luck searching the more narrowly defined indices.

How is search zone indexing set up? It depends on the search engine software used

Most support the creation of search zone, but some provides interfaces that make this process easier, while other require you to manually provide a list of pages to index. You can create search zones in many ways. Examples of four common approaches are:

- By content type
- By audience
- By subject
- By date

4.12.2 To Search or Not To Search

It's becoming a doubtful question whether to apply a search engine in your site. Users generally expect searching to be available, certainly in large sites. Yet we all know how poorly many search engine actually work. They are easy to set up and easy to forget about. That's why it's important to understand how users information needs can vary so much and to plan and implement your searching system's interface and search zones accordingly.

To find the answer to the question: TO SEARCH OR NOT TO SEARCH?

We need to know the need of Searching and searching Systems, advantages disadvantages of Searching and searching systems. All of these points have already been discussed.

4.13 Grouping Content

Grouping content into the top-level categories of an information hierarchy is typically the most important and challenging process you will face. How should the content be organized? By audience or format or function? How do users currently navigate this information? How do the clients want users to navigate? Which content items should be included in which major categories? The design of information architectures should be determined by research involving members of the team and representatives from each of the major audiences. Fortunately, you don't need the least technology to conduct this research. Index cards, the 3 x 5 inch kindly you can fit in your pocket and find information any stationery store, will help you get the job done. For lack of a better name, we call this index card based approach content chunking. To try content chucking, buy a few packages of index cards and follow these steps:

1. Invite the team to generate a content wish list for the website on a set of index cards.
2. Instruct them to write down one content item per card.
3. Ask each member of the group or the group as a whole to organize cards into piles of related content items and assign labels to each pile.
4. Record the results of each and then move on to the next.
5. Repeat this exercise with representative members and groups of the organization and intended audiences.
6. Compare and contrast the results of each.

Analysis of the results should influence the information architecture of the web site.

This card based content chunking process can be performed corroboratively where people must reach consensus on the organization of information. Alternatively, individuals can sort the cards alone and record the results. The biggest problem with shuffling index cards is that it can be time consuming. Involving clients, colleagues and future users in the exercise and analyzing the sometimes confusing results takes time. Some of this content chunking can be accomplished through the wish list process as noted earlier. However, the major burden of content chunking responsibility often falls to the information architect in the conceptual design phase.

4.14 ARCHITECTURAL PAGE MOCKUPS

Information architecture blueprints are most useful for presenting a bird's eye view of the web site. However they to not work well for helping people to envision the contents of any particular page. They are also not straightforward enough for most graphic designers to work from. In Fact no single format perfect job of conveying all aspects of information architecture to all audiences. Because information architectures are multi dimensional, it's important to show them information multiple ways. For these reasons. Architectural page mockup are useful tools during conceptual design for complimenting the blueprint view of the site mockups are quick and dirty textual documents that show the content and links of major pages on the website. They enable you to clearly (yet inexpensively) communicate the implications of the architecture at the page level.

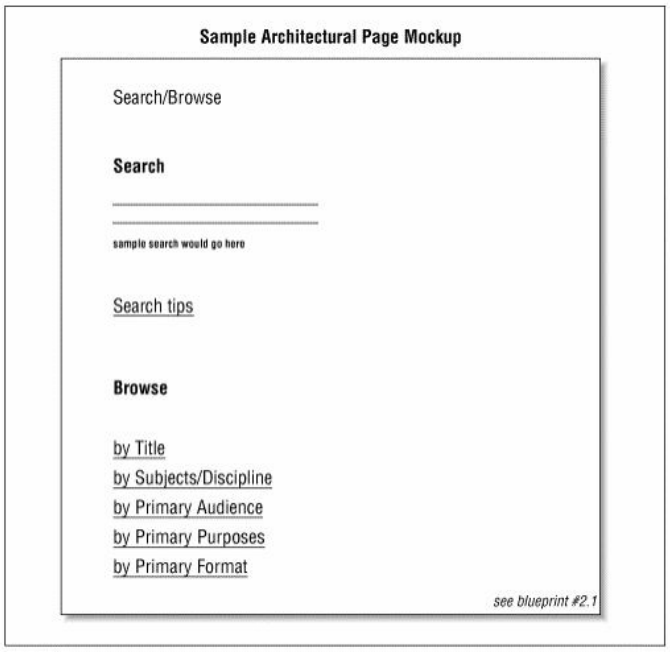


Fig. 4.8 : Architectural Mockup.

They are also extremely useful when used in conjunction with scenarios. They help people to see the site in action before any code is written. Finally, they can be employed in some basic usability tests to see if users actually follow the scenarios as you expect. Keep in mind that you only need to mockup major pages of the web site. These mockups and the designs that derive from them can serve as templates for design of subsidiary pages. The mockups are easier to read than blueprints. By integrating aspects of the organizational labeling, and navigation systems in to one view they will help your colleagues to understand the architecture. In laying out the content on a page mockup, you should try to show the logical visual grouping of content items. Placing a content group at the top of the page or using a larger font size indicates the relative importance of that content. While the graphic designer will make the final and more detailed layout decisions you can make a good start with these mockups.

In this figure, the architectural mockup of a combination search/browse page, we show an area for entering queries and an area for browsing. We typically use a word processor like Microsoft Word to create these mockups quickly. However, you can also create quick and dirty HTML mockups, and even work quite interactively with the graphic designer.

4.15 DESIGN SKETCHES

Once you have evolved high-level blueprints and architectural page mockups, and you are ready to collaborate with your graphic designer to create deign sketches on paper of major pages in the web site. In the research phase the design team has begun to develop a sense of the desired graphic identity or look and feel. The technical team has assessed the information technology infrastructure of the organization and the platform limitations of the intended audiences. They understand what's possible with respect to features such as dynamic content management and interactivity. And of course the architect has designed the high-level information structure for the site.

Design sketches are a great way to pool the collective knowledge of these three teams in a first attempt at interface design for the top level pages of the site. This in a wonderful opportunity for interdisciplinary user interface design using the architectural mocks ups as a guide; the designer begins sketching pates of the site on sheets of paper. As the designer sketches each page questions arise that must be discussed.

Here is a sample sketching session dialog:

- **Programmer:** I like what you're doing with the layout of the main page, but I'd like to do something more interesting with the navigation system.
- **Designer:** Can we implement the navigation system using pull down menus? Does that make sense architecturally?
- **Architect:** That might work but it would be difficult to show context in the hierarchy. How about a tear-|way table of contents feature? We've had pretty good reactions to that type of approach front users in the past.
- **Programmer:** We can certainly go with that approach from a purely technical perspective. How would a tear away table of contents look? Can you sketch it for us? I'd like to do a quick and dirty prototype. These sketches allow rapid iteration and intense collaboration. As you can see, the design of these sketches requires the involvement of people from all three teams. It is much cheaper and easier for the group to work with the designer on these rough sketches than to begin with actual HTML page layouts and graphics. These sketches allow rapid iteration and intense collaboration. The final product of a sketching session might look something like that in Figure.

In this example shown by the figure, Employee Handbook, Library, and News are grouped together as the major areas of the web site. Search/Browse and Guidelines/ Policies make up the bottom of the page navigation bar. A news area defines space for a dynamic Java-based news panel.

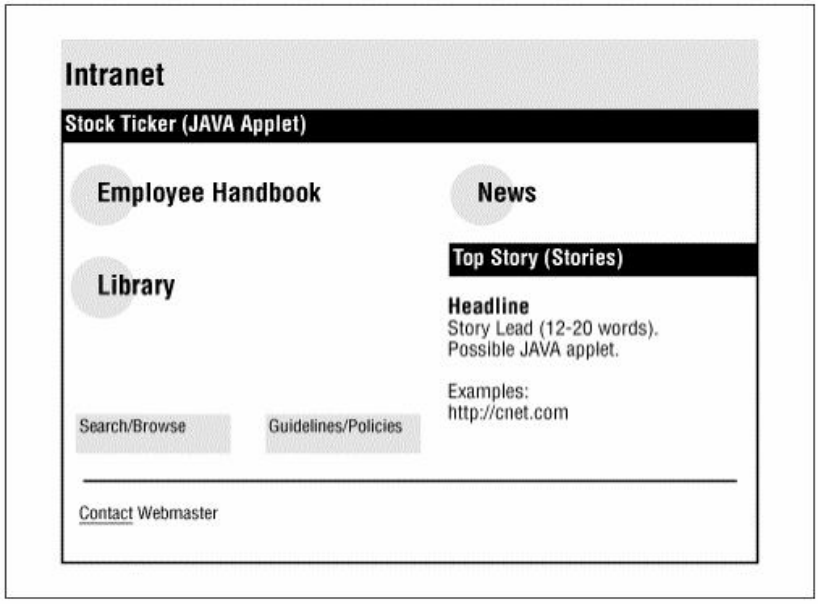


Fig. 4.9 : Design Sketches

SUMMERY

1 Navigation and Searching both are used for finding information. Navigation searches for the information to be found by moving between links available. But information searching we give the information about the information to be found as text to the search engine and search engine does the task of finding information for users.

2. As the amount of information on the website increases it become difficult to find the required information. If the navigation systems are not properly designed and maintained then to find the required information searching systems are required.

3. Searching only works well when the stuff that's being searched is the same as the stuff that users want. This means you may not want to index the entire site. Search engines are frequently used to index an entire site without regarded for the content and how it might vary.

4. Grouping content into the top-level categories of an information hierarchy is typically the most important and challenging process you will face. How should the content be organized? By audience or format or function? How do users currently navigate this information? How do the clients want users to navigate? Which content items should be included in which major categories? The design of information architectures should be determined by research involving members of the team and representatives from each of the major audiences.

5. Blueprints are the architect's tool of choice for performing the transformation for chaos in to order. Blueprints show the relationship between pages and other content components and can be used to portray organization, navigation and labeling systems.

6. Architectural page mockup are useful tools during conceptual design for

complimenting the blueprint view of the site mockups are quick and dirty textual documents that show the content and links of major pages on the website.

7. Design sketches are a great way to pool the collective knowledge of design and technical teams in a first attempt at interface design for the top level pages of the site.

SHORT QUESTION ANSWER

Q.1 What do you understand by search engine? What are the different components of search engine? Explain.

Ans. A search engine is an information retrieval system designed to help find information stored on a computer system. Search engines help to minimize the time required to find information and the amount of information which must be consulted, akin to other techniques for managing information overload.

A search engine consists of three components operates in the following order:

1. Web crawling 2. Indexing 3. Searching

Q.2 What are meta tags? How does meta tag help search engines in cataloging the information? Explain.

Ans: Meta tags are the HTML or XHTML <meta … > elements used to provide structured metadata about a Web page. Multiple elements are often used on the same page: the element is the same, but its attributes are different. Meta elements can be used to specify page description, keywords and any other metadata not provided through the other head elements and attributes.

Q.3 What do you understand by web site navigation systems? Where should it occur on a web page and why? What are the different types of navigation systems? Write a detailed note.

Ans. To look for any information atom we need to search for its link information its group. This is done using browsing/navigation user can get lost in the information space but a well-designed taxonomy may reduce the chances that user will become lost. So generally Navigation Systems are beneficial if information is organized using the hierarchy model. Complementary navigation tools are often needed to provide context and to allow for greater flexibility.

Types of Navigation Systems :

A) Embedded / integrated Navigation Systems: Embedded Navigation Systems are typically wrapped around and infused within the content of the site.

1. **Global (site-wide) Navigation System:** By definition, a global navigation system is intended to be present on every page throughout a site
2. **Local Navigation Systems:** Local Navigation Systems enable users to explore the immediate area.
3. **Contextual Navigation system:** The creation of contextual navigation

links specific to a particular page, document or object. E.g. Words or phrases within sentences are represented as embedded or inline hypertext links.

B) Supplemental/ Remote Navigation System: These navigation systems are external to the basic hierarchy of a website and provide complementary ways of finding content and completing tasks.

1. *Sitemaps:* In a book/ magazine, the table of contents presents the top few levels of the information hierarchy.
2. *Site Indexes:* Similar to the back of book index found in many print materials, a web based index presents keywords organization phrases alphabetically, without representing the hierarchy.
3. *Guides:* Guides take several forms including guided tours, tutorials and micro portal focused around a specific audience topic or task.

Q.4 How navigation system is designed for the web ? Explain.

Ans. Designing navigation systems that work well is challenging. You've got so many possible solutions to consider and lots of sexy technologies such as pop-up menus and dynastic site maps can distract you from what's really important: building context, improving flexibility, and helping the user to find the information they need. No single combination of navigation elements works for all web sites. One size does not fit all. Rather, you need to consider the specific goals, audience, and content for the project at hand, if you are to design the optimal solution.

Navigation can be divided into 3 categories :

1. Main navigation: Main Navigation should appear on all pages in the same style and in the same place. Choosing to place it on the left or the top is generally a good idea since this is where people expect to find it.
2. Subsidiary Navigation: Navigation specific to the section of the site page is in. Having subsidiary menus that only appears once a section, or mini sites is entered, can successfully combined navigation and location indicator.
3. Secondary Navigation: Secondary Navigation is important for at least three reasons:
   a. If the user lost, then a drop down box, search function or site map provides an opportunity for them to quickly reoriented them or get back to a familiar page.
   b. A site map page can provide more detail about what is in each section of the site than a navigation bar can.
   c. Secondary navigation provides a fast way for user to get content at a deep level, especially in areas of the site where there is a lot of content.

Q.5 What are the different methods of searching information on a web site?

Ans. Assuming you have decided to implement a Searching system for your website. It's important to understand how users really search before designing it. To illustrate

let's look at one of these factors in greater detail: The variability information users searching expectations.

1. Known item searching : Some users information needs are clearly defined and have a single correct answer. When you check the newspaper to see how your stock information amalgamated shoelace and aglet is 'doing (especially since the hostile Microsoft takeover attempts), you know exactly what you want that the information exists and where it can be found.

2. Existence searching : However some users know what they want but do not know how to describe it or weather the answer exists at all e.g., you must want to buy shares information Moldovan high start-ups and that carries no load.

3. Exploratory searching : Some users know how to phrase their question, but don't know exactly what they are hoping to find and are really just exploring and trying to learn more.

4. Comprehensive Searching (Research) : Some users want everything available on a given topic. Scientific researchers, patent lawyers, doctoral students trying to find unique and original dissertation topics, and fans of any sort fit in to this category.

Q.6 Write short note on Design Sketches.

Ans. Once you have evolved high-level blueprints and architectural page mockups, and you are ready to collaborate with your graphic designer to create deign sketches on paper of major pages in the web site. In the research phase the design team has begun to develop a sense of the desired graphic identity or look and feel. The technical team has assessed the information technology infrastructure of the organization and the platform limitations of the intended audiences.

Design sketches are a great way to pool the collective knowledge of these three teams in a first attempt at interface design for the top level pages of the site. This in a wonderful opportunity for interdisciplinary user interface design using the architectural mocks ups as a guide; the designer begins sketching pates of the site on sheets of paper.

- Programmer
- Architect
- Designer
- Programmer

Q.7 How website is planned ?

Ans. It is the first most important part of building a website. Before designing a website, it is necessary to do proper planning as in the case of building a house. If you start building your house without giving a thought of planning, the house may end up into an uncomfortable dwelling. Same way, for building a good website certain things should be taken care of. The following “Things to consider” while planning a website:

- Purpose of website
- Target audience

- Website contents
- Blue print of a website
- Directory structure of the website • Budgeting

Q.8 What are different Integrated navigation elements ?

Ans. In global and local navigation systems, the most common and important navigation elements are those that are integrated into the content-bearing pages of the web site. As users move through the site or sub-site, these are the elements they see and use again and again. Most integrated navigation elements fit into one of two categories: navigation bars and pull-down menus.

Navigation Bars : A navigation bar is a collection of hypertext links grouped together on a page. Alternatively, the navigation bar may be graphical in nature, implemented as an image map or as graphic images within a table structure. Graphic navigation bars tend to look nicer but can significantly slow down the page loading speed (although, if you're able to reuse the same global navigation bar throughout the site, loading speed will only be hurt once, since the image will be cached locally).

Pull-Down Menus : Pull-down menus compactly provide for many navigation options. The user can expand what appears as a single-line menu to present dozens of options (as shown in Figure 4.3). The most common pull-down menus on the Web are implemented using the standard interactive forms syntax. Users must choose an option from the menu and then hit a Go or Submit button to move to that destination.

IMPORTANT QUESTIONS

Q.1 What do you understand by web site navigation systems? Where should it occur on web page and why?

Q.2 What are the different types of navigation systems? Write a detailed note.

Q.3 What do you mean by Searching system? Describe the process of searching the information from various Website.

Q.4 What do you mean by Blueprint?

Q.5 What do you mean by Architectural Page Mockups?

Q.6 Write short note on Design Sketches.

Q.7 How website is planed?

Q.8 What are different website design issues?

****

HTML (HYPTERTEXT MARKUP LANGUAGE)

IN THIS CHAPTER, YOU WILL LEARN

- ☞ Introduction to HYML (Hyper Text Markup Language)
- ☞ History of HTML
- ☞ HTML Basic Concepts
- ☞ Structure of HTML document
- ☞ Block Level ELements
- ☞ Text Level of Elements

5.1 AN INTRODUCTION TO HTML (HYPERTEXT MARKUP LANGUAGE)

HTML is a computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively easy to learn, with the basics being accessible to most people in one sitting; and quite powerful in what it allows you to create. It is constantly undergoing revision and evolution to meet the demands and requirements of the growing Internet audience under the direction of the World Wide Web Consortium, W3C, the organization charged with designing and maintaining the language. The full meaning of HTML is HyperText Markup Language.

- **HyperText** is the method by which you move around on the web - by clicking on special text called hyperlinks which bring you to the next page. The fact that it is hyper just means it is not linear - i.e. you can go to any place on the Internet whenever you want by clicking on links.
- **Markup** is what HTML tags do to the text inside them. They mark it as a certain type of text (italicised text, for example).
- **HTML is a Language** as it has code-words and syntax like any other language.

5.2 HISTORY OF HTML

A markup language combines text as well as coded instructions on how to format that text and the term "markup" originates from the traditional practice of 'marking up' the margins of a paper manuscript with printer's instructions. Nowadays, however, if you mention the term 'markup' to any knowledgeable web author, the first thing they are likely to think of is 'HTML'.

In the Beginning HTML-which is short for HyperText Markup Language- is the official language of the World Wide Web and was first conceived in 1990. HTML is a product of SGML (Standard Generalized Markup Language) which is a complex, technical specification describing markup languages, especially those used in electronic

document exchange, document management, and document publishing. HTML was originally created to allow those who were not specialized in SGML to publish and exchange scientific and other technical documents. HTML especially facilitated this exchange by incorporating the ability to link documents electronically using hyperlinks. Thus the name Hypertext Markup Language.

What is XHTML?

XHTML, or eXtensible HyperText Markup Language, is a later version of HTML that includes additional standards set forth by the international organization called the World Wide Web Consortium (W3C). Both HTML and XHTML can be understood by most browsers, but for increased accessibility to your Web pages, we recommend using XHTML when creating your Web pages.

5.3 HOW IS HTML USED?

The formatting "rules" (HTML tags) that are applied to content are interpreted by programs (browsers) designed to display Web pages as specified by the HTML.

Why is HTML important?

- HTML is the foundation of all Web pages.
- For the same reasons we learn how to perform mathematical operations long-hand before we use a calculator, learning HTML can be useful.
- Because HTML pages (Web pages) are really just text-files they can be created with a simple text-editor (Notepad, WordPad, TextPad, SimpleText, Write). HTML editors may not always be available, warranting the use of a text- editor.

HTML Rules :

- An HTML document is built upon ASCII text that has been marked up with tags looking like:

```
<TAGNAME>
and
</TAGNAME>
```

- The purpose of these tags is to specify an device-free .
- For example, first level headers would be marked as:

```
<H1>This is a First Level Header</H1>
```

How HTML works on the Web

a) Your Computer : The browser on your computer sends a request for an HTML document from a remote server using addresses called URLs (Uniform Resource Locators). When the data is located and returned, your browser displays the web page (text and graphics) according to the HTML tags in the document.

b) Connection to the Internet : A dial up modem in your computer or a direct high speed data transmission line connects your computer to an internet service provider.

c) Internet Service Provider : Your internet service provider is probably an internet web server and is connected to all the other computers on the web. Your web server sends your request for an HTML document and sends back the file to you.

d) Internet : The internet is a collection of web servers around the world. Each

server has a URL and will forward your request on until it reaches the server you are looking for. When the data is returned to you, it may travel a totally different route over different computers.

e) Remote server : The remote web server with the URL you are looking for has all the HTML files including text, graphics, sound, and video. It may also have gateway scripts that are programs running on the server to process data.

5.4 HTML BASIC CONCEPTS

Before you start writing code to write a web page, it is a good practice to plan ahead the appearance of the web page. An HTML document has two elements:

1. Document Content 2. Elements

Document content is the information on a web page that the user will see. That information could be text or graphics, for example. As you start creating your own web pages, try finding out first what information you want to display and how you want to present it.

Elements are the structures that describe parts of an HTML document. Elements are the HTML codes that control how the document content will appear. For example, the **P** element represents a paragraph while the **EM** element gives *emphasized* content.

An element has three parts: a start tag, content, and an end tag. A **tag** is special text--"markup"--that is delimited by "**<**" and "**>**". An end tag includes a "**/**" after the "**<**". For example, the **EM** element has a start tag, <**strong**>, and an end tag, <**/strong**>. The start and end tags surround the content of the **EM** element :

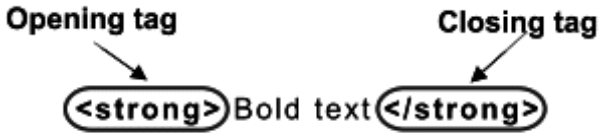


Fig. 5.1

Note: Element names are always case-insensitive, so <**strong**>, <**sTRONG**>, and <**STRONG**> are all the same.

An element's attributes define various properties for the element. For example, the font element takes a face attribute to provide the different font style:

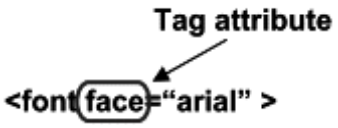


Fig. 5.2

An attribute is included in the start tag only--never the end tag--and takes the form ***Attribute-name="Attribute-value"***. The attribute value is delimited by single or double quotes. The quotes are optional if the attribute value consists solely of letters in the range A-Z and a-z, digits (0-9), hyphens ("-"), periods ("."), underscores ("_"), and colons (":").

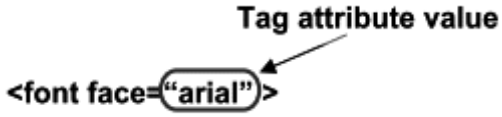


Fig. 5.3

Note: Attribute names are case-insensitive, but attribute values may be case-sensitive.

Empty HTML Elements :

HTML elements without content are called empty elements. Empty elements can be closed in the start tag.

 is an empty element without a closing tag (it defines a line break). Adding a slash to the start tag, like
, is the proper way of closing empty elements, accepted by HTML, XHTML and XML. Even if
 works in all browsers, writing
 instead is more future proof.

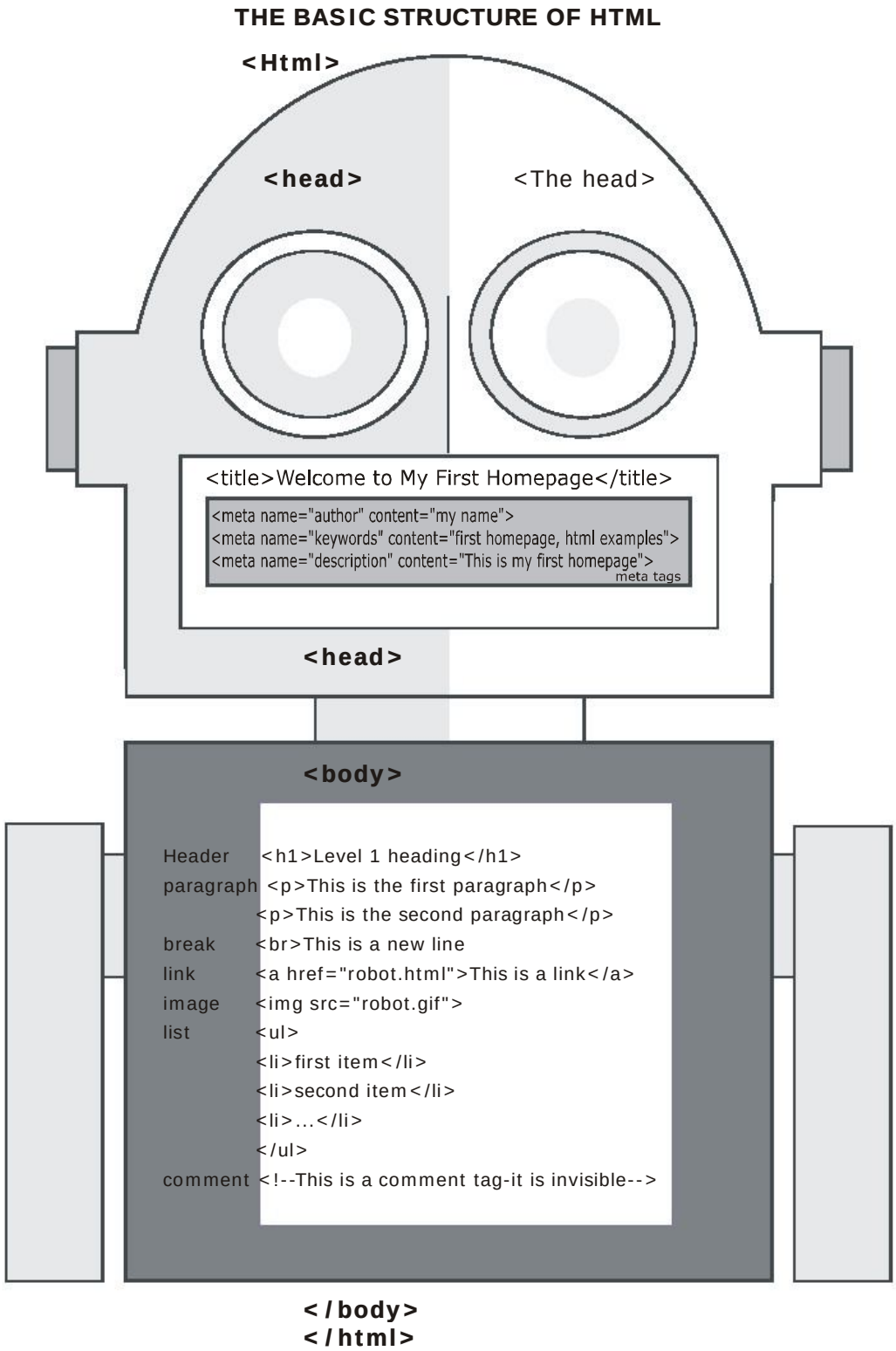


Fig. 5.4 : The Structure of HTML.

5.5 STRUCTURE OF HTML DOCUMENT

At the foundation of every HTML file is a set of structure tags that divide an HTML file into a head section and a body section. These two sections are enclosed between an opening <HTML> tag and end with the closing </HTML> tag. Following is a simple example of an HTML file that highlights the structure tags that are required within every XHTML file.

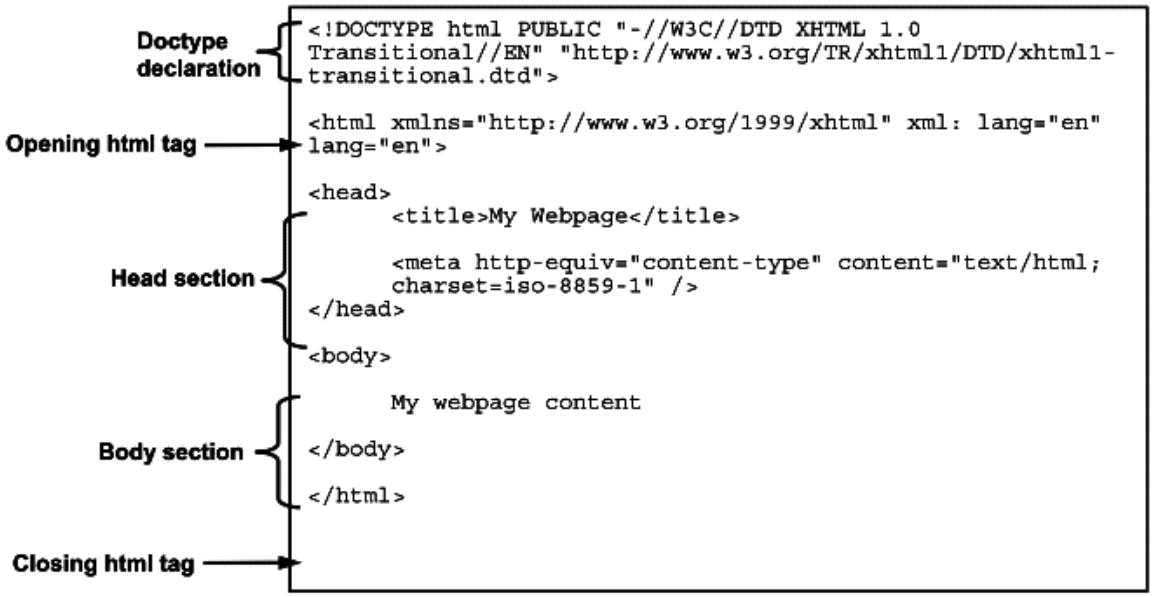


Fig. 5.5

DOCTYPE Declaration : The DOCTYPE declaration is the first part of coding that you should enter in your HTML document. This is required if you wish to validate your document with the W3C's validation service. Web browsers need to know what version of HTML/XHTML your page is written in to process the code correctly.

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

The HTML Tags : All HTML documents contain a <html> and </html> pair of tags. These tags identify the document's contents as HTML to the browser. The <html> tag goes in the line right under your DOCTYPE declaration. </html> is the last line of coding in your document.

Opening html tag:

```
<html>
```

The Head Tags

The <head> and </head> tags identify the document's head area. The information between these two tags is not visible on your page.

Opening head tag:

```
<Head>
```

The Title Tags

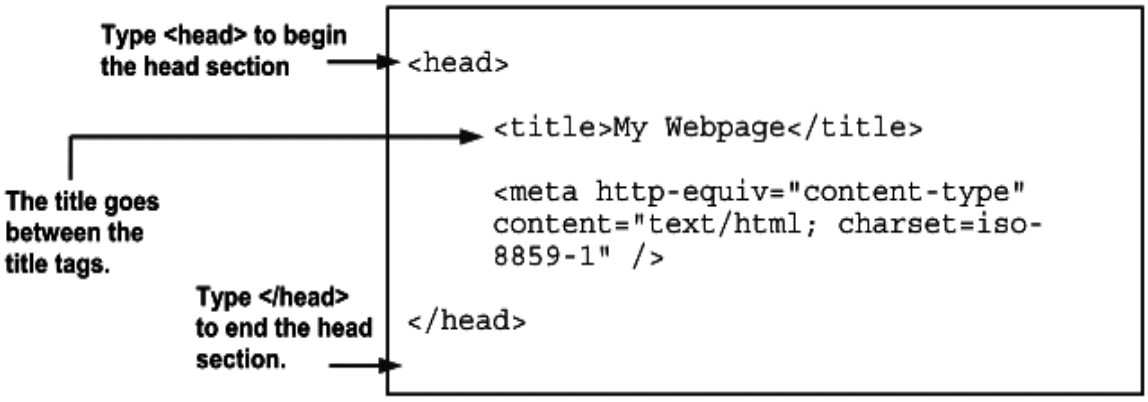


Fig. 5.6

The title tag creates the page title that is seen in the title bar of the web page.

```
<Title>My Web Page</Title>
```

Meta Tag

The meta tags provide information about your web page.

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
```

In addition to the title element, the head may also contain:

1. base : defines base URLs for links or resources on the page, and target windows in which to open linked content

2. link : refers to a resource of some kind, most often to a style sheet that provides instructions about how to style the various elements on the web page

3. Meta : provides additional information about the page; for example, which character encoding the page uses, a summary of the page's content, instructions to search engines about whether or not to index content, and so on

4. Object : represents a generic, multipurpose container for a media object

5. Script : used either to embed or refer to an external script

6. Style : provides an area for defining embedded (page-specific) CSS styles

All of these elements are optional and can appear in any order within the head. Note that none of the elements listed here actually appear on the rendered page, but they are used to affect the content on the page, all of which is defined inside the body element.

Closing Head Tag

The closing head tag defines the end of the document's head section.

```
<Head>
```

The Body

The body tags surround the body (contents) of your web page.

```
<Body>
The Body of the HTML Document
</Body>
```

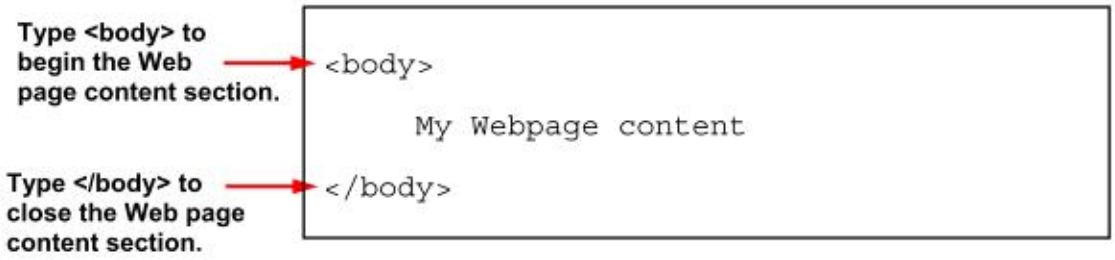


Fig. 5.7

Closing HTML Tag : The closing HTML tag is the last line in your HTML document. Don't put anything after this tag! Your page will not validate if you do.

```
</Html>
```

5.5.1 Core Attributes

HTML 4.0 has a set of four core attributes: ID, CLASS, STYLE and TITLE attribute.

1. ID Attribute: The ID attribute uniquely identifies an element within a document. No two elements can have the same ID value in a single document. The attribute's value must begin with a letter in the range A-Z or a-z, digits (0-9), hyphens ("-"), underscores ("_"), colons (":"), and periods (".") . The value is case-sensitive.

The following example uses the ID attribute to identify each of the first two paragraphs of a document:

```
<P ID=firstparagraph>This is my first paragraph.</P>
<P ID=secondparagraph>This is my second paragaph.</P>
```

In the example, both paragraphs could have style rules associated with them through their ID attributes. The following Cascading Style Sheet defines unique colors for the two paragraphs:

```
<style type="text/css">
#firstparagraph
{
font-weight:bold;
}
#secondparagraph
{
font-style:italic;
}
</style>
```

Output :

This is my first paragraph.

This is my second paragraph.

Note : A style sheet rule could be put in the head i.e.(<head>..<style>...</style> ...</head>) of a document.

2. Class Attribute: The class attribute is used to indicate the class or classes that a tag might belong to. Like id, class is used to associate a tag with a name, so

```
<p id="FirstParagraph" class="important">
This is the first paragraph of text.
</p>
```

not only names the paragraph uniquely as FirstParagraph, but also indicates that this paragraph belongs to a class grouping called important. The main use of the class attribute is to relate a group of elements to various style sheet rules. For example, a style sheet rule such as

```
<style type="text/css">
.important {background-color: yellow;}
</style>
```

would give all elements with the class attribute set to important a yellow background. Given that many elements can have the same class values, this may affect a large portion of the document.

3. Style Attribute : The style attribute is used to add style sheet information directly to a tag. For example,

```
<p style="font-size: 18pt; color: red;">
This is the first paragraph of text.
</p>
```

sets the font size of the paragraph to be 18 point, red text. Although the style attribute allows CSS rules to be added to an element with ease, it is preferable to use id or class to relate a document-wide or linked style sheet.

4. Title Attribute : The title is used to provide advisory text about an element or its contents. In the case of the

```
<p title="Introductory paragraph">
This is the first paragraph of text.
</p>
```

title attribute is set to indicate that this particular paragraph is the introductory paragraph. Browsers can display this advisory text in the form of a Tooltip, as shown here:

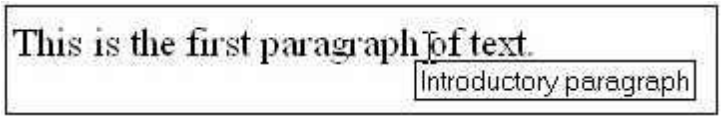


Fig. 5.8

Tooltips set with title values are often found on links, form fields, images, and

anywhere where an extra bit of information is required. The core attributes might not make a great deal of sense at this time because generally they are most useful with scripting and style sheets, but keep in mind that these four attributes are assumed with every tag that is introduced for the rest of this chapter.

5.5.2 Language Attributes

One major goal of HTML 4 was to provide better support for languages other than English. The use of other languages might require that text direction be changed from left to right across the screen to right to left. Nearly all HTML elements now support the dir attribute, which can be used to indicate text direction as either ltr (left to right) or rtl (right to left). For example:

```
<p dir="rtl"> This is a right to left paragraph. </p>
```

Furthermore, mixed-language documents might become more common after support for non-ASCII-based languages is improved within browsers. The use of the lang attribute enables document authors to indicate, down to the tag level, the language being used. For example,

```
<p lang="fr">C'est Francais. </p>
<p lang="en">This is English</p>
```

Although the language attributes should be considered part of nearly every HTML element, in reality, these attributes are not widely supported by all browsers and are rarely used by document authors.

5.5.3 Core Events

The last major aspect of modern markup initially introduced by HTML 4 was the increased possibility of adding scripting to HTML documents. In preparation for a more dynamic Web, a set of core events has been associated with nearly every HTML element.

Most of these events are associated with a user doing something. For example, the user clicking an object is associated with an onclick event attribute. So, would associate a small bit of scripting code with the paragraph event, which would be triggered when the user clicks the paragraph. In reality, the event model is not fully supported by all browsers for all tags, so the previous example might not do much of anything.

```
<!DOCTYPE html>
<html>
<body>
<h1 onclick=”this.innerHTML=’Ooops!’”>Click on this text</h1>
</body>
</html>
```

List of events

Below there is al list of all events available for the HTML 4.01 and XHTML 1.0 standards.

- **onload**: The onload event is triggered when the user agent finishes loading a window or all frames within a frameset. This event is exclusive of the HTML body element and the HTML framset element.
- **onclick**: The onclick event occurs when the mouse button is clicked over the element.
- **onmouseover**: The onmouseover event occurs when the mouse is moved onto the element.
- **onmousemove**: The onmousemove event is executed when the pointing device is moved while it is over the element.
- **onfocus**: The onfocus event is fired when an element receives focus either by the pointing device or by tabbing navigation. This attribute is exclusive of those elements that can get the focus.
- **onkeypress**: The onkeypress event is executed when a key is pressed and released while the element is focused. This attribute is exclusive of those elements that can get the focus: HTML a element, HTML area element, HTML label element, HTML input element, HTML select element, HTML textarea element, and HTML button element.
- **onkeydown**: The onkeydown event is triggered when a key is pressed down while the element is focused.
- **onkeyup**: The onkeyup event is fired when a key is released while the element is focused.
- **onsubmit**: The onsubmit event occurs when the form is submitted. This attribute is exclusive of those elements that can get the focus. This event is exclusive of the HTML form element.
- **onselect**: The onselect event is triggered when a user selects some text in the text field. This event is exclusive of the HTML input element and the HTML textarea element.
- **onchange**: The onchange event is fired when a control losses the input focus and its value has been modified since gaining focus. This event is exclusive of the HTML input element, the HTML select element and the HTML textarea element.

The content of the event is the code to be executed, and must be created using a cleint-side language (e.g., JavaScript) that must be supported by the browser. In the next example, we define a paragraph that changes the color of its text to red when the mouse passes over, and to black when it goes away.

5.6 Block-Level Elements

A block-level element is one that contains a significant block of content that should be displayed on its own line, to break apart long passages of text into manageable portions such as paragraphs, headings, and lists. Many nonempty, block-level elements can contain other block-level elements, and all can contain text and inline elements.

As with most word processors, HTML includes several tags to delimit, and hence,format

paragraphs of text. These tags include the following:

```
<p>-Formatted paragraphs
<h1>through <h6>-Headings
<blockquote>-Quoted text
<pre>-Preformatted text
<div>-A division of the document
<center>-Centered text
<BR>Line Break, <HR> Horizontal Rule
<ul>,<ol>, <dl>-Unnumbered, ordered, and de?nition lists
```

Each of the block elements results in a line break and noticeable space padding after the closing tag. As such, the block elements only work when used to format paragraph-like chunks of text-they cannot be used as inline tags.

5.6.1 FORMATTED PARAGRAPH

The paragraph tag p is used, quite simply, to mark up paragraphs of text:

Example:

```
<p> This is a paragraph of text……</p>
<p>…and this is another paragraph.</p>
```

The p element is one of the most commonly used building blocks of HTML. When you use the p element to begin a new paragraph in HTML, it automatically creates a line space above and below the content. This element may contain any text content, but it can't include any block-level elements: only inline or phrase elements can be included. The P element has an attribute called align that will center the content inside the two paragraphs.

Example:

```
<HTML>
<HEAD><TITLE>Paragraph Example</TITLE>
</HEAD>
<BODY>
<P>This is the first paragraph in the example about the P tag. There really isn't much to say here. </P>
<P ALIGN="center"> This is the second paragraph. Again, more of the same. This time the paragraph is aligned in the center. This might not be such a good idea as it makes the text hard to read. </P>
<P ALIGN="right"> Here the paragraph is aligned to the right. Right-aligned text is also troublesome to read. The rest of the text of this paragraph is of little importance. </P>
<P ALIGN="justify"> Under HTML 4.0-compliant browsers, you are able to justify text. As you may notice, the way browsers tend to justify text is sometimes imprecise. Furthermore, not all browsers support this attribute value. </P>
```

```
</BODY>
</HTML>
The align attribute affects the contents of the p, aligning content to
the "left", "right", or "center", or setting it to "justify" on the page.
```

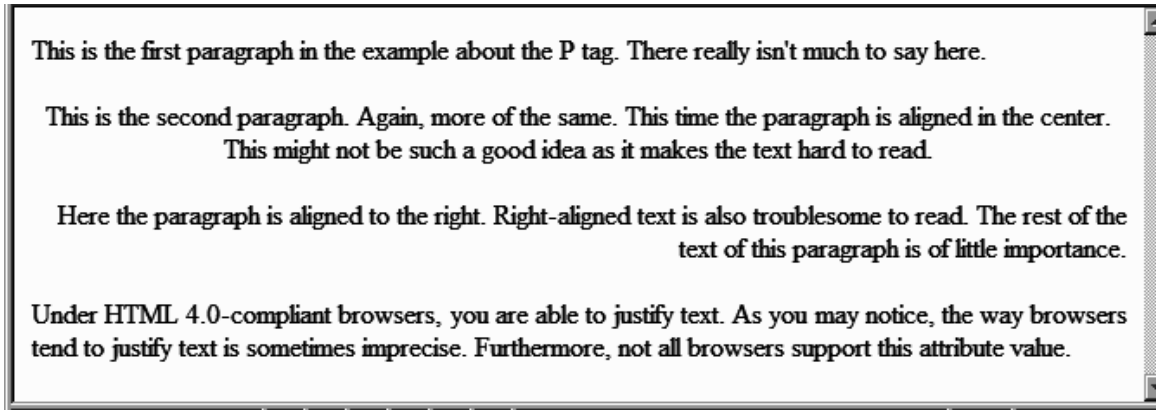


Fig. 5.9

5.6.2 Headings

The heading elements are used to create "headlines" in documents. Six different levels of headings are supported: <H1>, <H2>, <H3>, <H4>, <H5>, and <H6>. These range in importance from <H1>, the most important, to <H6>, the least important. Each heading uses a large, usually bold character-formatting style to identify itself as a heading. HTML automatically adds an extra blank line before and after a heading.

```
<html>
<body>
<h1> Heading 1 </h1>
<h2 ALIGN="center"> Heading 2 </h2>
<h3 ALIGN="right"> Heading 3 </h3>
<h4> Heading 4 </h4>
<h5> Heading 5 </h5>
<h6> Heading 6 </h6>
<p>Plain body text:
The quick brown fox jumped
over the lazy dog.</p>
</body>
</html>
```

An attribute that aligns the text left, right, or center can be added to the heading elements. By default, headings are usually left-aligned, but by setting the ALIGN attribute of the various heading elements, the text may be aligned to the right, left, or center of the screen.

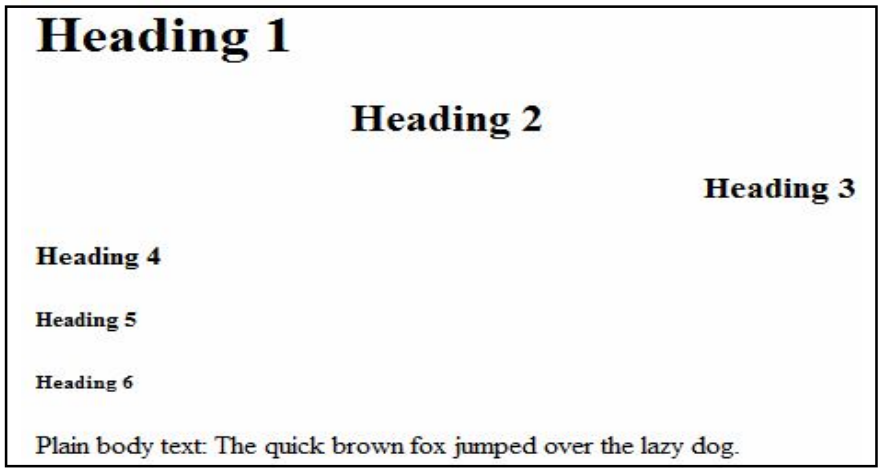


Fig. 5.10

5.6.3 Quoted Text

The blockquote tag gives you the option of setting off a long quotation or note that might otherwise get lost within a paragraph of text. This tag indents the entire selection on both the right and the left, and also adds a blank line above and below. The browser determines the exact amount of the indentation, and it may vary from browser to browser. Though the element is logical in nature, enclosing text within <BLOCKQUOTE> and </BLOCKQUOTE> usually indents the blocked information.

```
<p>Renowned type designer, Matthew Carter, has this to say about his
profession:</p>
<blockquote>
<p>Our alphabet hasn't changed in eons; there isn't much latitude in
what a designer can do with the individual letters.</p>
</blockquote>
<p>Much like a piece of classical music, the score is written down -
it's not something that is tampered with–and yet, each conductor
interprets that score differently. There is tension in the
interpretation.</p>
```

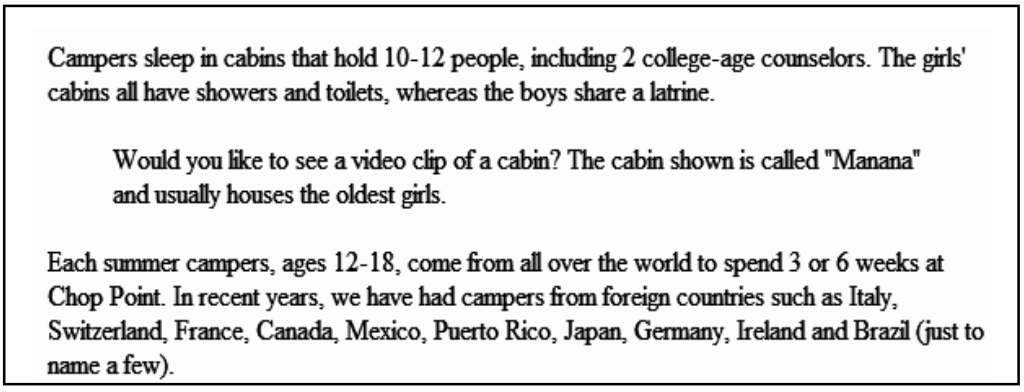


Fig. 5.11 : Shows the default rendering of the Blockquote example.

5.6.4 Divisions Tag

The <DIV></DIV> container (DIV stands for division) can be used to enclose and define the alignment for an entire block of page elements. It supports the ALIGN attribute, so you could use it to align a block of text and graphics to CENTER, as in this example:

```
<DIV ALIGN=CENTER>
<H1>This header is centered.</H1>
<P > This paragraph is also centred.</P><BR>
</DIV>
<DIV ALIGN=Right>
<P >  Many paragraph and other block elements can be affected by a DIV
at once. </P > <BR>
<P > Notice all the paragraphs are right aligned.</P>
</DIV>
```

Note : that all the elements between the <DIV> and </DIV> tags are aligned according to the definition given by the <DIV> tag, except for any elements which have their own alignments defined. As is always the case with the ALIGN attribute, it can assume the values LEFT, CENTER, or RIGHT.

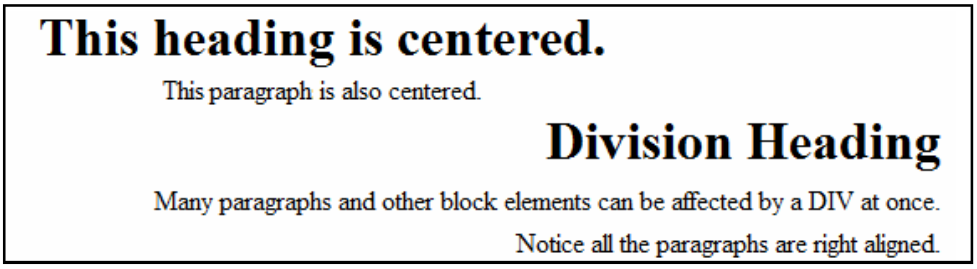


Fig. 5.12

5.6.5 The <CENTER> as a Block Element

In the original HTML 2-based browsers, centering text was impossible. One of the major additions introduced by HTML 3.2 was the <CENTER> element because of its widespread use. To center text or embedded objects (such as images), simply enclose the content within <CENTER> and </CENTER>. In this sense, <CENTER> appears to be a text-formatting style element, but under the HTML 3.2 and transitional 4.0 specifications (and beyond), <CENTER> is defined as an alias for a block-level structuring element. Under the HTML 4.0 DTD, <CENTER> is simply an alias for <DIV ALIGN="CENTER"> and is treated exactly the same way.

```
<BODY>
<CENTER>
<H1>This heading is centered.</H1>
<P>This paragraph is also centered.</P>
<P>Many paragraphs and other block elements
can be affected by a CENTER at once.</P>
</CENTER>
</BODY>
</HTML>
```

The <CENTER> element is unlikely to go away, considering its simplicity and widespread use. But according to specifications, two preferred ways exist to center content: the <DIV> element with a center alignment attribute, or the ALIGN attribute used in conjunction with some elements.

5.6.6 Preformatted Text

The <PRE> and </PRE> tags can be used to surround text that shouldn't be formatted by the browser. The text enclosed within the <PRE> tags retains all spacing and returns, and doesn't reflow when the browser is resized. Scroll bars and horizontal scrolling are required if the lines are longer than the width of the window. The browser generally renders the preformatted text in a monospaced font, usually Courier. Some text formatting, such as bold, italics, or links, can be used within the <PRE> tags.

The pre element in this example displays as shown in Figure 5-3. The second part of the figure shows the same content marked up as a paragraph (p) element for comparison.

```
<pre>
This is     an          example of
    text with a             lot of
                        curious
                      white space.
</pre>   <p>
This is a example of text with a lot of curious white space.
</p>
```

```
This is              an            example of
      text with a          lot of
                          curious
                          white space.
```

Fig. 5.13:The spaces & returns remain intact when the content is rendered.

5.6.7 Horizontal Rule

One way you can separate sections of your Web page is to use the hr tag. By default, this tag produces a thin, gray horizontal line called a horizontal rule. The <HR> element is an empty element, because it has no close tag and encloses no data, the W3C recommends using <hr/> to officially start and end this tag. Several attributes can change the appearance of the horizontal rules you use on your pages such as SIZE sets the bar's thickness (height). WIDTH sets the bar's width. ALIGN sets its vertical alignment. NOSHADE renders the bar without a surrounding shadow.

```
<HR SIZE=4 WIDTH="50%">
```

Fig. 5.14

Line Breaks :

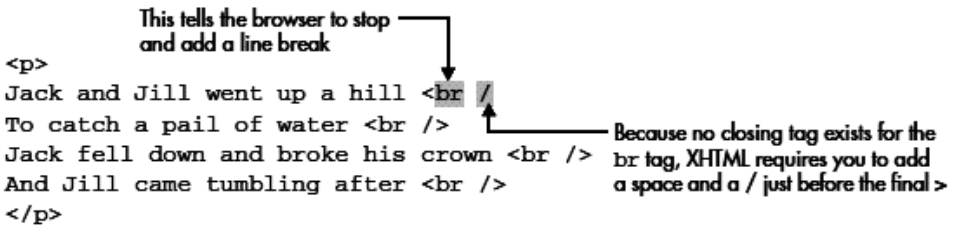


Fig. 5.15

The
 tag is used to add a line break in your HTML page. It causes the browser to stop printing text on that line and drop down to the next line on the page. Because the
 tag has no end tag (or closing tag), it breaks one of the rules for future HTML (the XML based XHTML), namely that all elements must be closed. Writing it like
 is a future proof way of closing (or ending) the tag inside the opening tag, accepted by both HTML and XML.

5.6.9 HTML Lists

There may be times when web designers need to organize information or images in a neat format resembling a list. This allows the designer to group related pieces of items together so they can be clearly seen increasing readability and importance. HTML has tags for this purpose and supports the following three block-level list types: **1. Unordered List 2. Ordered List 3. Definition Lists.**

5.6.9.1 Unordered Lists : Unnumbered or unordered lists are usually displayed with bullets, which depict each new line of information to be displayed in the list structure. Unordered lists should be used when needing to display a related group of data without stressing the order in which the data is presented. The following code illustrates an example of an unordered list:

```
<html>
<head><title>Unordered List Example</title></head>
<body>
 <ul>
<li>List Item 1</li>
<li>List Item 2</li>
<li>List Item 3</li>
<li>List Item 4 </li>
</ul>
</body>
</html>
```

NOTE : The type="" attribute can be used in the opening <ul> tag to specify a square, circle, or disc shaped bullet. If no bullet is specified, the default solid disk shape is used. Also note that the closing tag is optional when entering <li> items.

5.6.9.2 Numbered Lists: Numbered lists are coded identical to an unnumbered list except for the opening and closing tag, which are: <ol> . Numbered lists

should be used when needing to display data when order is important.

```
<html>
<body>
<ol type="A">
<li>List Item 1
<li>List Item 2
<li>List Item 3
<li>List Item 4
</ol>
 </body>
</html>
```

NOTE: The following is a listing of the possible numbering styles for the type="" attribute:

VALUE	MEANING
1.	Arabic (1, 2, 3, ...) [default]
(a)	Alphabetic uppercase
(a)	Alphabetic lowercase
2.	Roman numeral uppercase
(a)	Roman numeral lowercase

5.6.9.3 Definition Lists: Definition lists can be used for two purposes. If needing to list terms with definitions, designers can use alternating definition tags <dt> and definition data tags <dd>. Note that using closing tags when using the <dt> and <dd> tags is optional. Designers can also use a definition list when using a custom bullet image instead of the standard bullet types of numbered and unordered lists. If using a custom bullet, only <dd> opening and closing tags should be used to list the bullet images and items. To begin and end a definition list, use the <dl></dl> start and closing tags. The following examples illustrate the two types of definition lists:

```
<html>
<body>
<dl>
<dt><b><i>Word 1</i></b></dt>
<dd>This corresponds to the meaning of word 1.</dd><br><br>
<dt><b><i>Word 2</i></b></dt>
<dd>This corresponds to the meaning of word 2.</dd><br><br>
<dt><b><i>Word 3</i></b></dt>
<dd>This corresponds to the meaning of word 3.</dd><br><br>
</dl>
</body>
</html>
```

The above HTML code would produce the following results in a web browser:

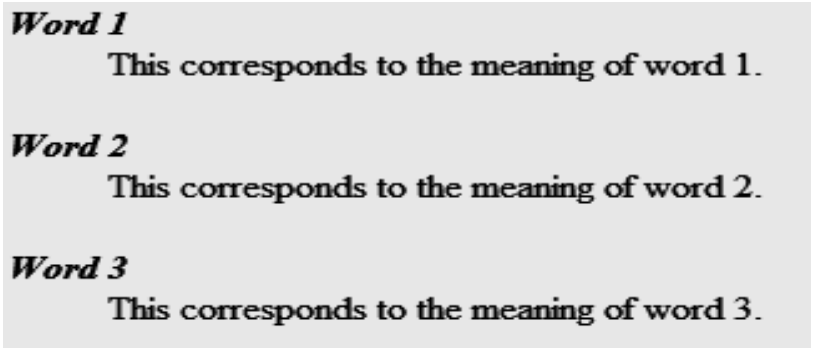


Fig. 5.16

5.7 TEXT-LEVEL ELEMENTS

These Text-Level Elements are also called phrase elements because they're intended to wrap around a short string of a few words, or even a single word, to give it added meaning and formatting that sets it apart from the other words that surround it. Text elements in HTML come in two basic flavors: physical and logical.

Physical elements, such as <B> for bold and <I> for italic, are used to specify how text should be displayed. Logical elements, such as <STRONG> and <EM>, indicate what text is, but not necessarily how it should look.

5.7.1 Physical Character-Formatting Elements : HTML supports various elements that can be used to influence physical formatting. They have the same effect in all the browsers such as Netscape Navigator, Internet Explorer and Mozilla.

1. <b> Element: Anything that appears in a <b> element is displayed in bold. Note: This <b> element has the same effect as the element, which you will meet later, and is used to indicate that its contents have strong emphasis.

```
This is in <B>bold</B> text.
```

2. <i> Element: The content of an <i> element is displayed in italicized text. The <i> element has the same effect as the element, which you will meet later, and which is used to indicate that its contents have emphasis.

```
This is in <I>italic</I> text.
```

3. <u> Element: The content of a <u> element is underlined with a simple line:

   This text is <U>underlined</U>.

4. <s> and <strike> Element: The content of an <s> or <strike> element is displayed with a thin line through the text (<s> is just the abbreviated form of <strike>).

```
This is a <STRIKE>strikethough</STRIKE> example.
```

5. The <tt> Element: The content of a <tt> element is written in monospaced font.

```
This is in TT>teletype</TT> text.
```

6. The <sup> Element: The superscript element moves the text higher than

the surrounding text and (if possible) displays the text in a smaller size font.

This is a ^{superscript}.

7. The <sub> Element: The subscript element moves the text lower than the surrounding text and (if possible) displays the text in a smaller size font.

   This is a _{subscript}.

8. The <big> Element: The content of the <big> element is displayed one font size larger than the rest of the text surrounding it. If the font is already the largest size, it has no effect. You can nest several <big> elements inside one another, and the content of each will get one size larger for each element.

   This is <BIG>big</BIG> text.

9. The <small> Element: The content of the <small> element is displayed one font size smaller than the rest of the text surrounding it. If the font is already the smallest, it has no effect. You can nest several <small> elements inside one another, and the content of each gets one size smaller for each element.

   This is <SMALL>small</SMALL> text.

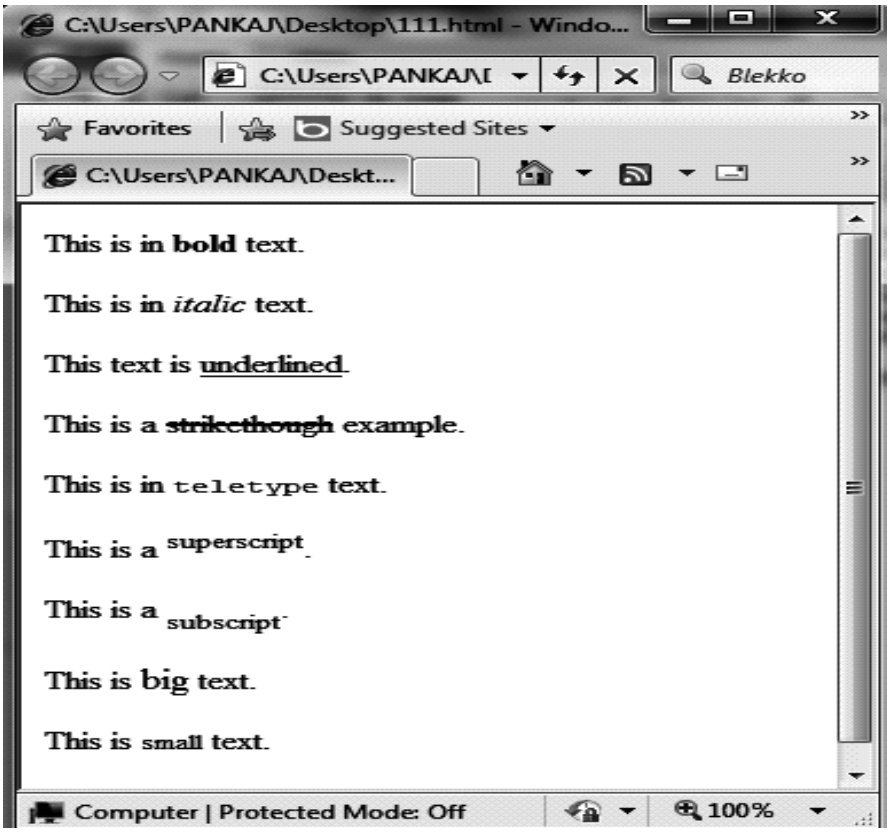


Fig. 5.17 : Shows the use of the all physical tags.

5.7.2 Logical Elements (Phrase Elements) : The semantic text elements describe the enclosed text's meaning, context or usage. The way they look

when they appear in the browser window depends on a style sheet, either one you provide or the browser's built-in default rendering.

Element	Description
abbr	abbreviation
acronym	acronym
cite	citation; a reference to another document, such as a book title
code	program code sample
del	deleted text; indicates an edit made to a document
dfn	the defining instance or first occurrence of a term
em	emphasized text
ins	inserted text; indicates an insertion in a document
kbd	keyboard; text entered by a user (for technical documents)
q	short, inline quotation
samp	sample output from programs
strong	strongly emphasized text
var	a variable or program argument (for technical documents)

Adding emphasis to text : There are two elements that indicate that text should be emphasized: em for emphasized text and strong for strongly emphasized text. Emphasized text elements almost always display in italics by default.

```
<p>Garamond is a <em>really</em> popular typeface, but Times is a
<strong>really really</strong> popular typeface.</p>
```

Garamond is a *really* popular typeface, but Times is a **really really** popular typeface.

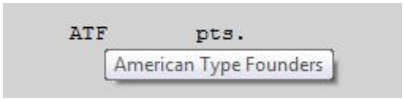
Short Quotations : Use the quotation (q) element to mark up short quotations, such as "To be or not to be" in the flow of text, as shown in this example.

Matthew Carter says, <q>Our alphabet hasn't changed in eons.</q>

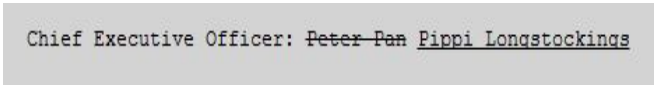
Matthew Carter says, "Our alphabet hasn't changed in eons."

Abbreviations and Acronyms : Marking up shorthand terms as acronyms and abbreviations provides useful information for search engines, screen readers, and other devices. Abbreviations, indicated by the abbr element, are shortened versions of a word ending in a period (Conn. for Connecticut, for example). Acronyms, indicated by the acronym element, are abbreviations formed by the first letters of the words in a phrase (such as WWW or USA). Both elements use the title attribute to provide the long version of the shortened term, as shown in this example.

```
<acronym title="American Type Founders">ATF</acronym>
<abbr title="Points">pts.</abbr>
```



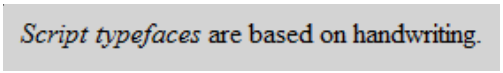
Citations : The cite element is used to identify a reference to another document, such as a book, magazine, article title, and so on. Citations are typically rendered in italic text by default. Here's an example:




```
<p>Passages of this article were inspired by <cite>The Complete Manual
of Typography</cite> by James Felici.</p>
```

Defining terms : The <dfn> element allows you to specify that you are introducing a special term. Its use is similar to the words that are in italics in the midst of paragraphs in this book when new key concepts are introduced.

```
<p><dfn>Script typefaces</dfn> are based on handwriting.</p>
```



Program Code elements : A number of inline elements are used for describing the parts of technical documents, such as code (code), variables (var), program samples (samp), and user-entered keyboard strokes (kbd).

Code, sample, and keyboard elements typically render in a constant-width (also called monospace) font such as Courier by default. Variables usually render in italics.

The browser displays the content in code element as given below in the following figure.

Inserted and deleted text : The ins and del elements are used to mark up changes to the text and indicate parts of a document that have been inserted or deleted (respectively).

Chief Executive Officer: <del title="retired">Peter Pan</del><ins>Pippi Longstockings</ins>

The FONT Element : The method that HTML uses for providing control over the appearance of the text is the FONT element. The FONT element is a container that is opened with the <FONT> start tag and closed with the </FONT> end tag. Unless attributes are assigned in the start tag, there is no effect of using a FONT element.

The FONT element can be used inside of any other text container and it will modify the text based upon the appearance of the text within the parent container. Using the FACE, SIZE, and COLOR attributes, you can use FONT to drastically modify the appearance of text in your documents.

1. **The FACE Attribute:** The FACE attribute allows you to specify the font that you would like the viewing software to use when displaying your document. The parameter for this attribute is the name of the desired font.

2. **The SIZE Attribute:** The SIZE attribute of the FONT element allows the document author to specify character height for the text. Font size is a relative scale from 1 though 7 and is based upon the "normal" font size being 3.
3. **The COLOR Attribute:** Text color can be specified in the same manner as the face or the size. The COLOR attribute accepts either a hexadecimal RGB value or one of the standard color names.

```
<HTML>
<HEAD>
<TITLE>Font Selection Example</TITLE>
</HEAD>
<BODY>
<FONT FACE="Verdana", "Arial", "Helvetica" SIZE=4 COLOR="#FF0000">
This is an example of font element with different fonts, size of 4 and color red. </FONT>
</BODY>
</HTML>
```

The <BASEFONT> Tag

The <BASEFONT> tag is used to establish the standard font size, face, and color for the text in the document. The choices made in the <BASEFONT> tag remain in place for the rest of the document, unless they are overridden by a FONT element. When the FONT element is closed, the BASEFONT characteristics are returned. BASEFONT attributes can be changed by another <BASEFONT> tag at any time in the document. Note that BASEFONT is a tag and not a container. There is no</BASEFONT> end tag.

```
<BASEFONT SIZE=6 FACE="GEORGIA">
```

SHORT ANSWER TYPE QUESTIONS

Q.1 What Exactly Is HTML?

Ans. The name HTML stands for Hypertext Markup Language. Many people who create Web pages and work in HTML often forget what the letters stand for. The term's hypertext portion refers to the cross-links, also called hyperlinks, between Web pages. The term's markup language portion refers to the commands that format the Web pages that the users see. Knowing how to write and use HTML is the goal, not remembering the archaic abbreviation.

Q.2 Define HTML filename extension.

Ans. A Web page, defined in an HTML file, always has the filename extension .html or .htm (if you want to be compatible with Windows 3x users, although fewer and Fewer of them exist). The html extension separates the file type from ordinary, unformatted text files whose extensions might be txt. Many browsers, such as Internet Explorer, will refuse to open your file with an extension such as txt, except by starting another program such as Notepad and loading the text file into that secondary program for your viewing and editing work. Some browsers will open a file whose name does not end with the html extension, but will refuse to interpret any HTML command tags. In such a case, the file will appear inside the browser window displaying the nitty-gritty command tags themselves instead of performing the formatting actions that the command tags request.

Q.3 What is Web Accessibility?

Ans. All Web pages are created in a standard format called hypertext markup language, or HTML for short. When you create or save a document as a web page, it is given a filename ending with .htm or .html, indicating that it is an HTML file. An HTML document actually contains nothing but plain text—images, sounds and other non-text elements are stored in separate files. You can look at the HTML code of any web page in your web browser:

- In Netscape Navigator, choose the View > Page Source command.
- In Internet Explorer, choose View > Source.

**Q.4 Define
 tag?**

Ans. Use the break tag to break lines. The format of the tag is as follows:
Text that appears on its own line. The
 tag is special because, unlike so many other command tags,
 has no corresponding end tag. The
 tag is a stand-alone tag because it requests that the browser move down to the next line on the screen before displaying the text that follows.

Q.5 Define HTML Headings.

Ans. HTML headings are defined with the <h1> to <h6> tags.

Headings Are Important: Use HTML headings for headings only. Don't use headings to make text BIG or bold. Search engines use your headings to index the structure and content of your web pages. Since users may skim your pages by its headings, it is important to use headings to show the document structure.H1 headings should be used as main headings, followed by H2 headings, then the less important H3 headings, and so on.

Example:

```
<h1>hello</h1>
<h2>hello</h2>
<h3>hello</h3>
<h4>hello</h4>
<h5>hello</h5>
<h6>hello</h6>
```

Q.6 Define HTML Paragraphs

Ans. HTML paragraphs are defined with the <p> tag.

Example:

```
<p>this is my first paragraph writing</p>
```

Q.7 How to use comment in html document?

Ans. HTML Comments-> Comments can be inserted into the HTML code to make it more readable and understandable. Comments are ignored by the browser and are not displayed.

Comments are written like this:

Example

```
<! – This is a comment –>
```

Q.8 How can we incorporate Horizontal Rule?

Ans. Horizontal Rule (HR): The HR element is a divider between sections of text; typically a full width horizontal rule or equivalent graphic. For example :

```
<HR>This creates an horizontal Line </HR>
```

Q.9 Explain Ordered and unordered list in html?

Ans. **HTML Unordered Lists**: An unordered list starts with the <ul> tag. Each list item starts with the <li> tag. The list items are marked with bullets (typically small black circles).

```
<ul>
<li>Coffee</li>
<li>Milk</li>
</ul>
```

How the HTML code above looks in a browser:

- Coffee
- Milk

HTML Ordered Lists: An ordered list starts with the <ol> tag. Each list item starts with the <li> tag. The list items are marked with numbers.

```
<ol>
<li>Coffee</li>
<li>Milk</li>
</ol>
```

How the HTML code above looks in a browser:

1. Coffee
2. Milk

HTML Definition Lists: A definition list is a list of items, with a description of each item. The <dl> tag defines a definition list. The <dl> tag is used in conjunction with <dt> (defines the item in the list) and <dd> (describes the item in the list):

```
<dl>
<dt>Coffee</dt>
<dd>- black hot drink</dd>
<dt>Milk</dt>
<dd>- white cold drink</dd>
</dl>
```

How the HTML code above looks in a browser:

Coffee
- black hot drink
Milk
- white cold drink

Q.10 How can we incorporate Horizontal Rule?

Ans. Horizontal Rule (HR) : The HR element is a divider between sections of text; typically a full width horizontal rule or equivalent graphic. For example :

```
<HR>This creates an horizontal Line </HR>
```

MULTIPLE CHOICE QUESTIONS

1. Is.htm and .html the same?

a) No b) Yes

2. Horizontal rule element is used to divide the webpage into:

a) Two sections b) Separate sections

c) Four sections d) All sections

3. Which tags are most commonly used by search engines?

a) Heading b) Title

c) Paragrah d) All of above

**4. What is the
 tag for?**

a) Space b) Paragraph break

c) Line break d) Word break

5. Html document must always be saved with:

a) .html b) .htm

c) .doc d) Both a &b

6. Logical elements indicate _____of the webpage.

a) Logical description b) General description

c) Both a &b d) None of these

7. Each list item in an ordered or unordered list has which tag?

a) list tag b) ls tag

c) li tag d) ol tag

8. How many types of lists are supported by html:

a) Two b) Three

c) Four d) Five

9. <meta> tag cannot be defined in the <head> tag.

a) False b) True

10. Which of the following attributes below are used for a font name?

a) fontname b) fn

c) font d) face

11. Is width="100" and width="100%" the same?

a) No b) Yes

12. What are tags used for?

a) To replace paragraphs. i.e. p tags

b) To logically divide the paragraphs

c) To logically divide the document

d) To provide space between tables

13. What is cell padding?

a) Used to separate cell walls from their contents.

b) Used to set space between cells

c) Both a and b above

d) Used to provide width to a cell

14. Can I play audios in HTML?

a) No b) Yes

KEY TO OBJECTIVE QUESTIONS

1.a	2.d	3.d	4.c	5.d	6.c
7.c	8.b	9.a	10.d	11.b	12.c
13.a	14.b				

IMPORTANT QUESTIONS

Q.1 What are the difference between Traditional HTML and Non-traditional HTML documents types ?

Q.2 Define HTML. What are the structure of HTML document ?

Q.3. What are the different types of tags available in HTML ? Explain with example.

Q.4 What is format of a good Web Page ?

Q.5 What do you mean by Tag ? Explain the following tags (with attributes) :

(a) <A> (b) <img>

(c) <meta> (d) <form>

(e) <Body> (f) <frame>

Q.6 What do you understand about DOCTYPE in HTML?

Q.7 How to use Line Break and Horizontal Line tags in HTML?

Q.8 What are Core Attributes and Core Events in HTML ?

****

6

WEB DESIGNING

IN THIS CHAPTER, YOU WILL LEARN

- Hyperlink
- Images
- Layout with Tables
- HTML Frames
- HTML AND Media Types
- Style Sheets
- Working With Forms

6.1 HYPERLINK

The acronym HTML stands for ***hypertext***, or text that is linked to other information. HTML enables us to link to other Web pages, as well as graphics, multimedia, e-mail addresses, newsgroups, and downloadable files. Anything you can access through your browser can be linked to from within an HTML document.

Web pages can contain links that take you directly to other pages and even specific parts of a given page. These links are known as ***hyperlinks***. A hyperlink is text, an image, or any other object in an HTML document that can be clicked in order to gain access to another web document. Normally, links are distinguished as being external or internal to the current page.

External web site link provides access to another web site that is not part of the current web site, such as going to *www.yahoo.com* from our current website mypage.html. An internal web site link provides access to another web page which is apart of the original web site, such as going to the next article in this guide from this web page. Finally, an internal document section link points to a region within a web page document.

Linking Documents : The <a> Element: A link is specified using the <a> element. This element is called anchor tag as well. Anything between the opening <a> tag and the closing tag becomes part of the link and a user can click that part to reach to the linked document. Following is the simple syntax to use this tag.

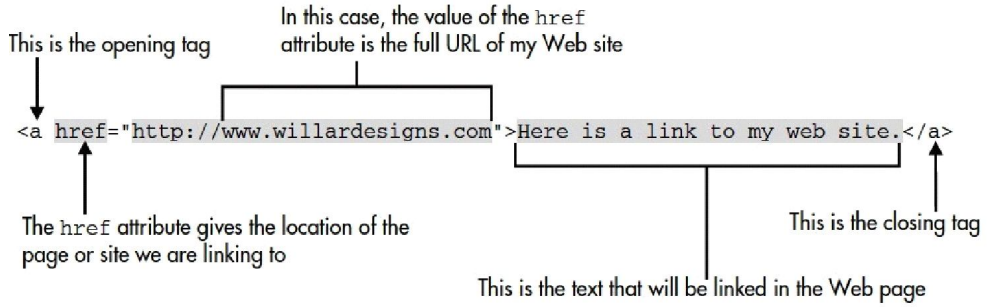


Fig. 6.1

6.1.1 External Linking

The <a> tag itself doesn't serve much purpose without its attributes. The most common attribute is href, which is short for hypertext reference: it tells the browser where to find the information to which you are linking. The text included in between the opening and closing a tag is what the person viewing your Web page can click. In most cases, this text is highlighted as a different color from the surrounding text and is underlined, as shown in Figure.

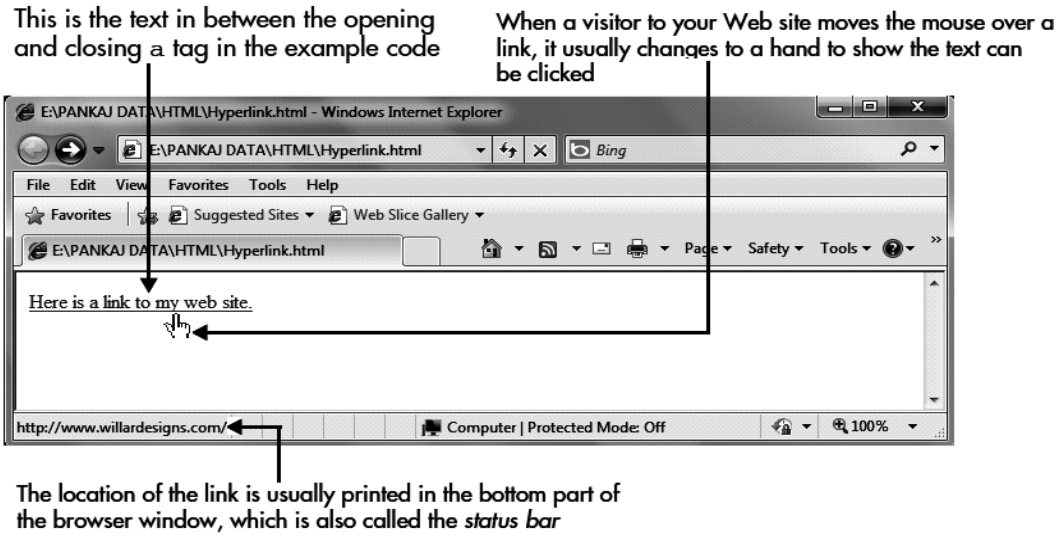


Fig. 6.2

Anchor Attributes : Following are most frequently used attributes for <a> tag.

- **href:** specifies the URL of the target of a hyperlink. Its value is any valid document URL, absolute or relative, including a fragment identifier or a JavaScript code fragment.
- **target:** specify where to display the contents of a selected hyperlink. If set to "_blank" then a new window will be opened to display the loaded page, if set to "_self" then loads the new page in current window. By default its "_self", if set to "_name" then open the link in the window of that name.
- **name & id:** attributes places a label within a document. When that label is used in a link to that document, it is the equivalent of telling the browser to goto that label.
- **event:** attributes like onClick, onMouseOver etc. are used to trigger any Javascript ot VBscript code.
- **title:** attribute lets you specify a title for the document to which you are linking. The value of the attribute is any string, enclosed in quotation marks. The browser might use it when displaying the link, perhaps flashing the title when the mouse passes over the link.

6.1.2 Internal Linking

Base Path for Links: It is not required to give a complete URL for every link. You can

get rid of it if you will use <base> tag in your header. This tag is used to give a base path for all the links. So your browser will concatenate given relative path to this base path and will make a complete URL.

For example we have used following base tag in all the pages at Ometasystems.com:

```
<head>
<base href="http://www.ometasystems.com/">
</head>
```

*So now if you will use <**a href="/html/index.htm"** then it will be considered as*

<a href="http://www.ometasystems.com/html/index.htm".

In deciding what to use as the value of your href attribute, consider what type of link you want to use. Two basic types of links exist.

- Absolute
- Relative

Absolute Links : Absolute links are those that include the entire pathname. In most cases, you use absolute links when linking to pages or sites that are not part of your own Web site. For example, if you are linking from your Web site to Yahoo!, you type "http://www.yahoo.com" as your link.

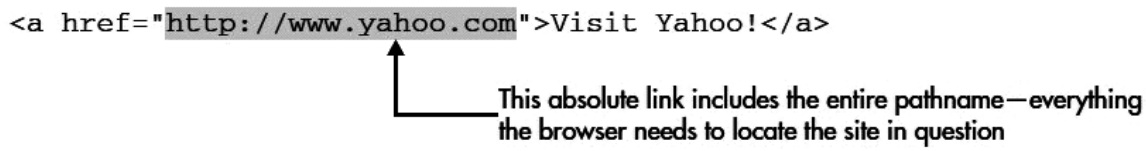


Fig. 6.3

Relative Links : Relative links are called so because you don't include the entire pathname of the page to which you are linking. Instead, the pathname you use is relative to the current page. Relative links are most commonly used when you want to link from one page in your site to another. Here's an example of what a relative link might look like:

```
<a href="contactme.html">Contact Me</a>
```

This link looks for the contactme.html file in the same folder that contains this page. If you were linking to a file in another folder below the current one, the value of your href might look like this:

```
<a href="wendy/contactme.html">Contact Me</a>
```

Links to Sections within the Same Page : Sometimes you may want to link to a section of text within a page on your Web site. To link to a section of a Web page, you must first give that section a name.

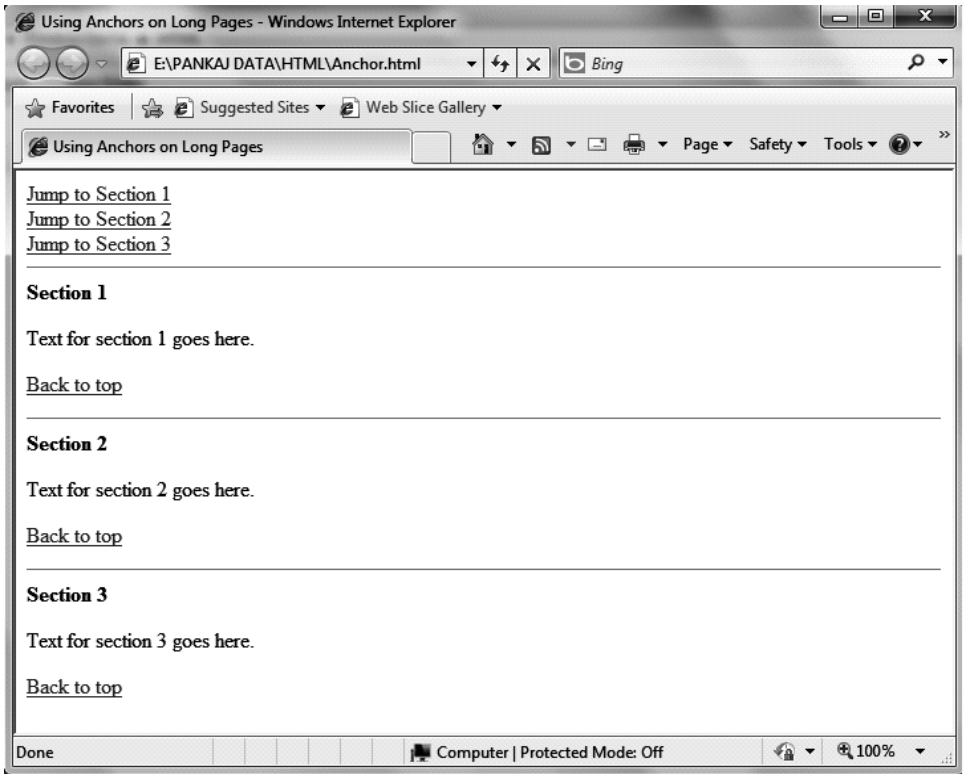


Fig. 6.4

Create an Anchor : An anchor is a place within a page that is given a special name, enabling you to link to it later. Without first naming a section, you cannot link to it. Here is an example of an anchor:

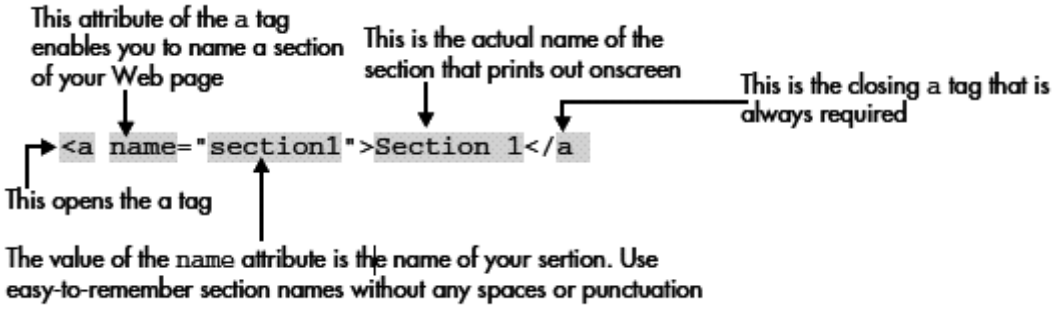


Fig. 6.5

In this example, the phrase in between the opening and closing a tag is displayed in the Web page and labels the anchor as "Section 1."

Link to an Anchor : To create the link to an anchor, you also use the a tag and the href attribute, as you would when creating any other type of link. To finish the link, you need to include a hash symbol (#) and the anchor name as the value of the href attribute.

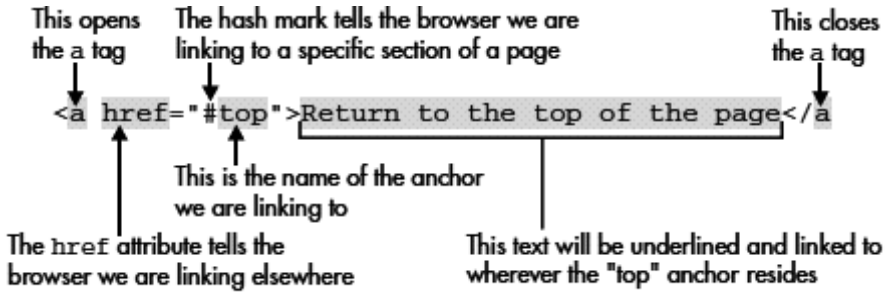


Fig. 6.6

Lets us create a program to link to different sections of the page.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/transitional.dtd">
<html>
<head>
     <title>Using Anchors on Long Pages</title>
</head>
<body bgcolor="#ffffff" text="#000000">
<a name="top"></a>
<a href="#section1">Jump to Section 1</a><br />
<a href="#section2">Jump to Section 2</a><br />
<a href="#section3">Jump to Section 3</a><br />

<hr />
<a name="section1"><b>Section 1</b></a>
<p>Text for section 1 goes here.</p>
<p><a href="#top">Back to top</a></p>
<hr />
<a name="section2"><b>Section 2</b></a>
<p>Text for section 2 goes here.</p>
<p><a href="#top">Back to top</a></p>
<hr />
<a name="section3"><b>Section 3</b></a>
<p>Text for section 3 goes here.</p>
<p><a href="#top">Back to top</a></p>
</body>
</html>
```

This links to the anchor lower on the page named "section1"

This links to the anchor lower on the page named "section 2"

This links to the anchor lower on the page named "section 3"

The anchor names this "section1"

The anchor names this "section2"

The anchor names this "section3"

Fig. 6.7

If you need to create a link to a specific section with another page (not the one you are currently working on), then you use that page's filename and the anchor name separated by a hash mark (#), as in the following example.

```
<a href="genealogy.html#intro">View names beginning with an "A" on our genealogy page.</a>
```

In this case, the browser will first look for genealogy.html and then locate an anchor named "intro" on that page.

Target Windows: Have you ever visited a Web site and noticed that a second instance of the Web browser opened when you clicked a link? This happens when Web developers use the target attribute to load links in a browser window other than the one you are currently using.

For example, you might want to offer visitors to your site a link to search Yahoo!, but you don't want to encourage them to leave your site. If you use "_blank" as the value of the target attribute in your link to Yahoo!, the browser will launch a new browser window to load http://www.yahoo.com.

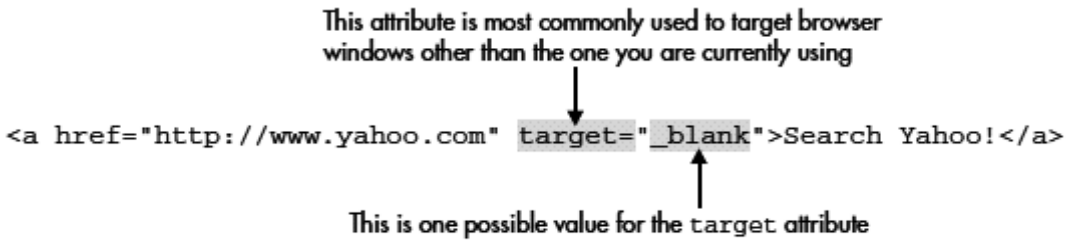


Fig. 6.8

Links to E-mail Addresses : When you want to give someone easy access to your e-mail address, you can include it on your page as a mailto link. This means instead of using http:// in front of your links, you use mailto: to preface youre-mail addresses. Clicking this link in a browser causes the visitor's e-mail program to launch. Then it opens a new e-mail message and places your e-mail address in the To: box of that message.

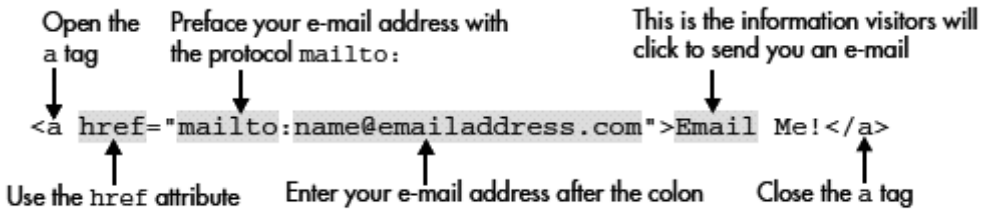


Fig. 6.9

Setting Link Colors: You can use three more attributes of the body tag to customize the three link colors of a Web page: normal link colors (link), visited link colors (vlink) and active link colors (alink).

```
<body bgcolor="#ffffff" text="#000000" link="#003366"
vlink="#999999" alink="#ff33cc">
```

Note : Default Link Colors: In most cases, the default link color for browsers is blue. The default visited link color is purple, and the active link color is red. Remember, as with many other features of Web browsers, the user ultimately controls these default colors.

6.1.3 Semantic Linking Meta Information

The Semantic Web is an evolving development of the World Wide Web in which the

meaning (semantics) of information and services on the web is defined, making it possible for the web to understand and satisfy the requests of people and machines to use the web content. It derives from World Wide Web Consortium director Sir Tim Berners-Lee's vision of the Web as a universal medium for data, information, and knowledge exchange.

At its core, the semantic web comprises a set of design principles, collaborative working groups, and a variety of enabling technologies. Some elements of the semantic web are expressed as prospective future possibilities that are yet to be implemented or realized. Other elements of the semantic web are expressed in formal specifications. Some of these include Resource Description Framework (RDF), a variety of data interchange formats (e.g. RDF/XML, N3, Turtle, N-Triples), and notations such as RDF Schema (RDFS) and the Web Ontology Language (OWL), all of which are intended to provide a formal description of concepts, terms, and relationships within a given knowledge domain.

6.2 IMAGES

So far in this book, you have created text-only Web pages. They're perfectly functional, but a bit dull. Web pages are more interesting and attractive when they include graphics. Much of the Web's popularity is due to its ability to present graphical content rather than only text, and even a few well placed graphics can make a huge difference in the look of a web page. Graphics of any size increase the time it takes to download a page, however, so you should use them with care to avoid making your pages awkward to access. Images appear on web pages in two ways: as part of the inline content or as tiling background images. Background images are added using background attribute in body tag or using Cascading Style Sheets. Inline images may be used on web pages in several ways:

- As a simple image. An image can be used on a web page much as it is used in print, as a static image that adds information, such as a company logo or an illustration.
- As a link. As we saw in the previous chapter, an image can be used as a link to another document by placing it in the anchor element.
- As an imagemap. An imagemap is a single image that contains multiple links ("hotspots") that link to other documents. We'll look at the markup used to add clickable areas to images in this chapter as well.

6.2.1 Image Preliminaries

The most common and most widely supported image file types are

GIF : The Graphics Interchange Format (GIF) was the earliest format use in inline images on the Web. GIF format has the ability to encapsulate several images within one file, giving the format animation functionality.

The GIF format has the following characteristics :

- Supports up to 8-bit color (256 colors)
- Supports transparency
- Is stored in a compressed, lossless format

- Can be interlaced and used for rudimentary animations

JPEG : JPEG is the acronym for Joint Photographic Experts Group and it was created specifically for photographic imagery and shouldn't be used for flat-color graphics. JPEG is a commonly used method of lossy compression for digital photography (image). With JPEG, you get to keep all your colors, but you don't get to keep all the data about the image.

- Supports 24-bit color (64,000 colors)
- Does not support transparency
- Is stored using a lossy compression format; the smaller the file, the lower the quality of the Image.
- Can be stored in a "progressive" format

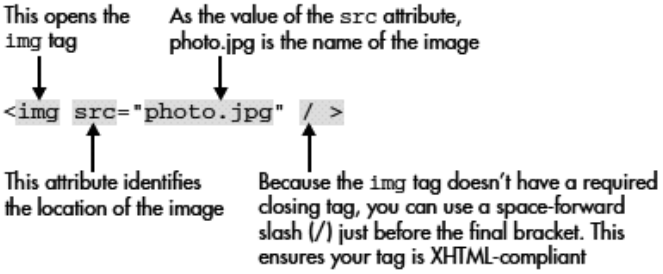
PNG : The Portable Network Graphics, or PNG format, was developed exclusively for the Web and is in the public domain. The PNG format takes advantage of a clever way of storing the information about the image so you don't lose color and you don't lose image quality; it is a lossless format. The only drawback is, because the standard is so new, only fourth-generation and later browsers support PNG graphics.

The PNG format has the following characteristics:

- Supports 24-bit color (64,000 colors)
- Supports transparency
- Is stored using a lossless compression scheme
- Can be stored and displayed in interlaced format

6.2.2 The Img Tag and SRC Attribute

In HTML, images are defined with the <img> tag. The img tag is empty, which means that it contains attributes only and it has no closing tag. To display an image on a page, you need to use the src attribute, src stands for "source". The value of the src attribute is the URL of the image you want to display on your page.

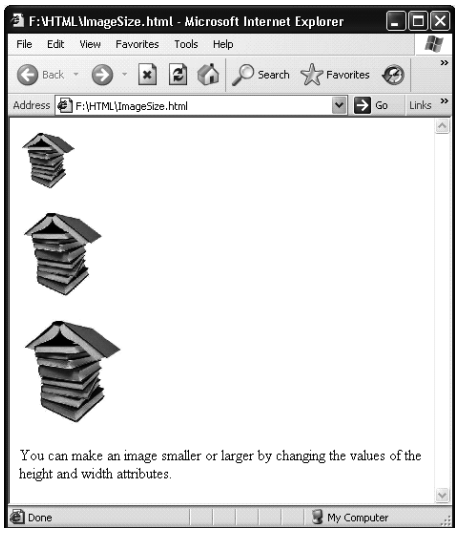


The syntax of defining an image : The URL points to the location or address where the image is stored. The form of the URL may be either an absolute URL or a relative URL.

6.2.3 Change the size of an Image

After you start adding several images to your Web pages, you may notice they sometimes cause the browser to wait a little while before displaying the page. Because they don't know the size of the image, some browsers actually wait until the images are all loaded before displaying the Web page. Therefore, you can help

speed the display of your Web pages by telling the browser the sizes of your images right within the img tag. You do so with the height and width attributes.



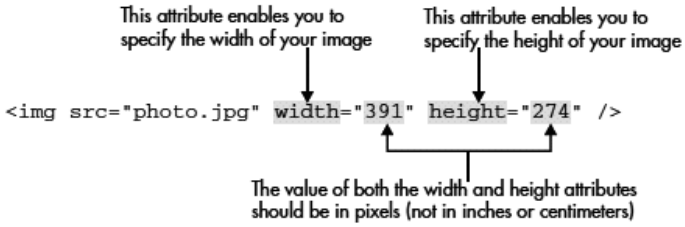


Fig. 6.10

The alt and title Attribute

The alt attribute is required to specify a text alternative for the image in case the user cannot see the image. Often referred to as alt text, it is important that the value of this attribute really describes the image. Two common reasons why images are not visible to users are:

1. Because the browser did not download the file correctly; the file cannot be found
2. Because the user has visual impairment that prevents him or her from seeing the image.

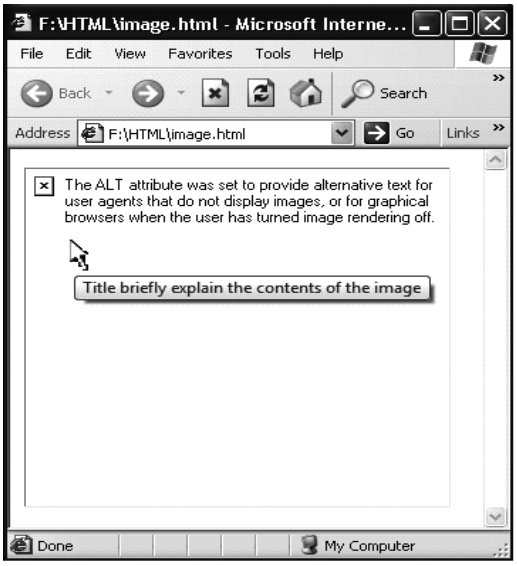
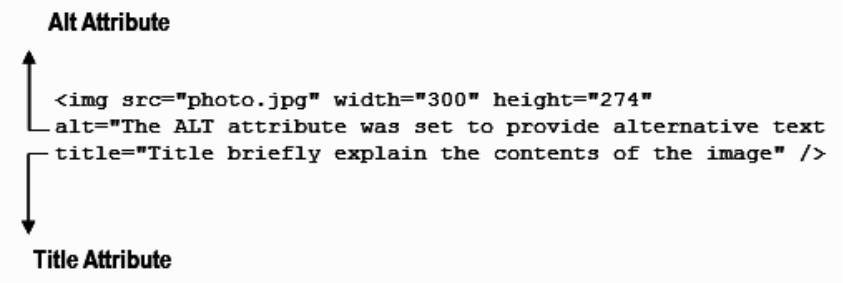


Fig. 6.11

Sometimes images do not convey any information, and are only used to enhance the layout of the page. (For example, you might have an image that is just a design element but does not add any information to the page.) Then the alt attribute should still be used but given no value, as follows:



In addition to the alt attribute, it's a good idea to add the title attribute to your img tag. While the alt attribute specifies alternative text for images in case the images don't load, the title attribute can be added to images as well as links and other page elements. It serves as a quick tip for users to briefly explain the contents of the

page element or, in this case, the image. The greatest benefit in using the title attribute is that its contents are displayed as "tool tips" in all browsers.

The border Attribute (deprecated) : To apply borders to a graphic, add the border attribute to the <img/> tag, and specify the appropriate properties.

```
<IMG SRC="book11.jpg" BORDER=0>
<IMG SRC="book11.jpg" BORDER=1>
<IMG SRC="book11.jpg" BORDER=2>
```

If the attribute is not used, there will not be a border unless the image is used as a link, in which case you could specify border="0"

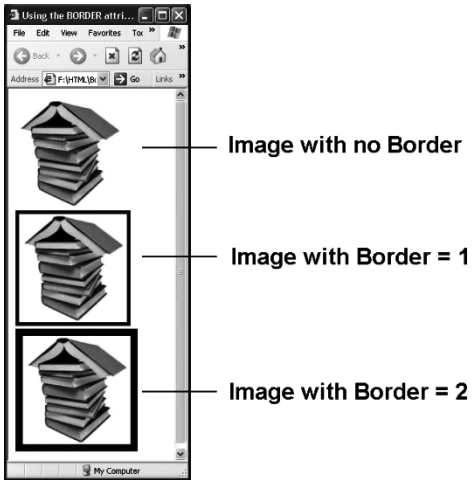


Fig. 6.12

6.2.4 Align the Image with an align attribute

By default, the browser displays the image inline. Thus, the browser displays it immediately to the right of any text or any other object that immediately precedes the image. Take a look at Figure, for example. It shows the same image three different times. Each time, the image is shown inline. That is, the browser displays the image immediately to the right of any text preceding it as seen below.

Aligning Text with an Inline Image : By default, when you insert an image inline with text, the text is aligned with the bottom of the image. You can change it, though, using the <IMG> tag's ALIGN attribute. Table describes each value you can assign to this attribute.

Value	Description
TOP	Aligns the text with the top of the image
MIDDLE	Aligns the text with the middle of the image
BOTTOM	Aligns the text with the bottom of the image

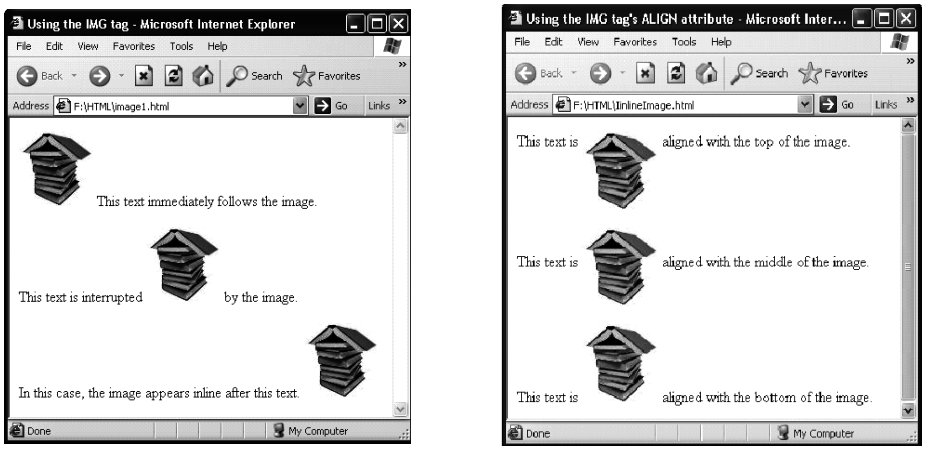


Fig. 6.13

6.2.5 Positioning an Image on the Web Page

By default, the browser displays images inline. That is, it displays an image immediately to the right of the previous content. Text does not wrap around it. You can display an image on the left or right side of the Web page, however, allowing the surrounding content to flow around the image. This type of image is called a floating image. You create a floating image by using the <IMG> tag's ALIGN attribute. This is the same attribute you use to align the surrounding text with an image. Table describes each value you can assign to this attribute.

```
<img src="book.gif" alt="My Book"
width="200" height="203" align="left" />

<img src="book.gif" alt="My Book"
width="200" height="203" align="right" />
```

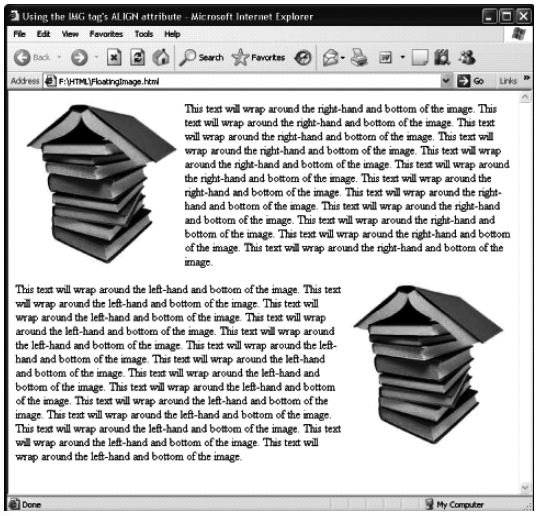


Fig. 6.14

Value	Description
LEFT	Displays image on left side and surrounding flows around the image
RIGHT	Displays image on right side of the window and surrounding flows around the image

**Importance of
 Tag :** In flowing text around an image, there may be a situation in which the designer wants to clear the text flow around the image.

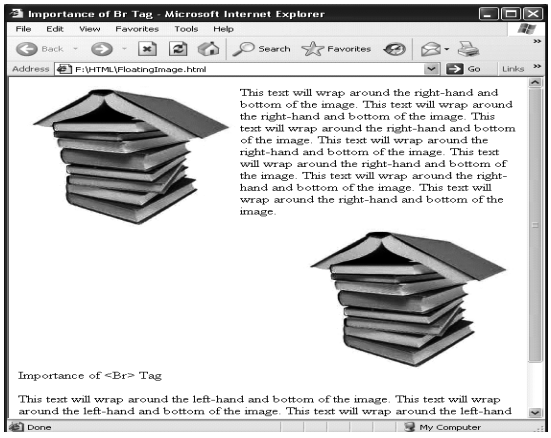
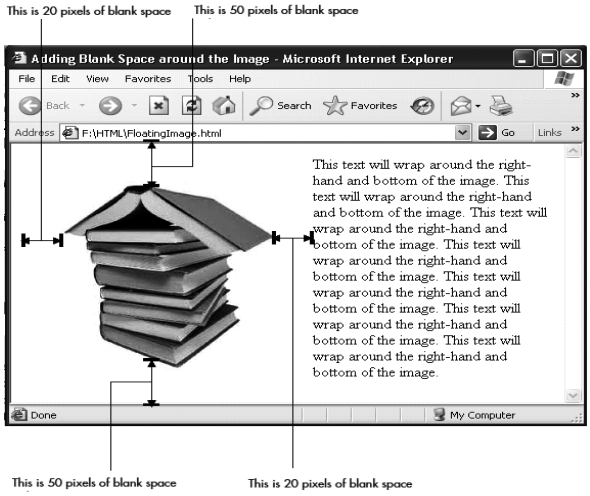


Fig. 6.15

To deal with such problems, a new attribute called **CLEAR** was added to the **
** Element, this attribute now part of the HTML standard. The **CLEAR** attribute can be set to **LEFT**, **RIGHT**, **ALL**, or **NONE** and will clear the gutter around an inline object like an image. For example, imagine the fragment <**IMG SRC="book. jpg" ALIGN="RIGHT"**> with text wrapping around it. If <**BR CLEAR="RIGHT"**> is included in the text and the wrapped text is still wrapping around the image, the text will be cleared to pass the image.

Add Space around Images : If you need to add some buffer space around an image, you can use the vspace and hspace attributes. The vspace attribute enables you to add vertical space above and below an image, while the hspace attributes add horizontal space to the right and left of an image. Both attributes require values in pixel dimensions.




```
<img src="book.gif" alt="My Book"
 width="200" height="203"
 align="left" hspace="20" vspace="50' />
```

Hspace and Vspace attribute to add space around the image

Fig. 6.16

6.2.6 Hyperlink your Image

In the previous chapter, you learned how to create text hyperlinks using the <a> tag. Recall that you place the URL in the opening <a> tag, and then you place the hyperlink text between the <a> and tags. You create a graphical hyperlink in much the same way, by placing an <img> tag in an <a> tag like this:

This opening a tag tells the browser to begin linking to the book.html page

This tag defines the image

```
<a href="book.html" ><img src="book.gif" width="391" height="274"
alt="My Book of Learning HTML" /> </a >
```

This closing a tag tells the browser to stop linking to the book.html page

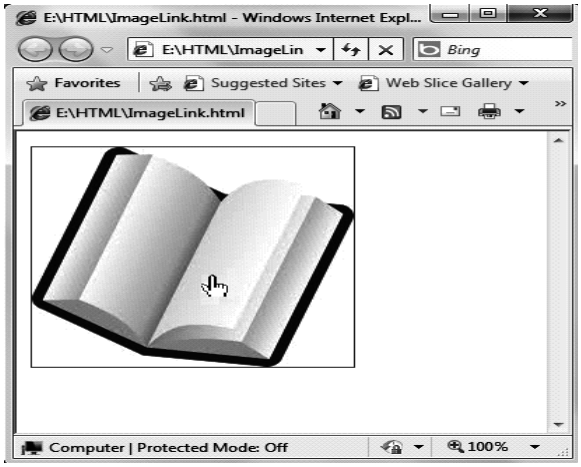


Fig. 6.17

6.2.7 Add a Background Graphic

Instead of placing a graphic inline with the text, you may want to use it to form the background of a web page. A background graphic can make your web pages much more colorful and dramatic, but you must make sure that the graphic neither obscures the text of the page nor clashes with any graphics you place inline. To add a background image, add the background attribute to the <body> tag, and specify the filename in double quotation marks.

```
<body background="picture.jpg">
```

Fig. 6.18

6.2.8 Image Maps

Image maps allow you to specify several links that correspond to different areas of one single image, so that when users click different parts of the image they get taken to different pages. There are two types of image maps:

- Server-side image maps,
- Client-side image maps

In the server-side image map, the user clicks on an image but the server must decode where the user clicked before the destination page (if any) is loaded. With client-side image maps, all of the map information, which regions map to which URLs can be specified in the same HTML file that contains the image. Including the map data with the image and letting the browser decode it.

It has several advantages, including :

1. There is no need to visit a server to determine the destination, so links are resolved faster.
2. Destination URLs can be shown in the status box as the user's pointer moves over the image.
3. Image maps can be created and tested locally, without requiring a server or system administration support.

4. Client-side image maps can be created so that they present an alternative text menu to users of text-only browsers.

Client-Side Image Maps using <map> and <area> : In this, the image that is going to form the map is inserted into the page using the <img/> element as normal, except it carries an extra attribute called usemap. The value of the usemap attribute is the value of the name attributes on the <map> element, which you are about to meet, preceded by a pound or hash sign.

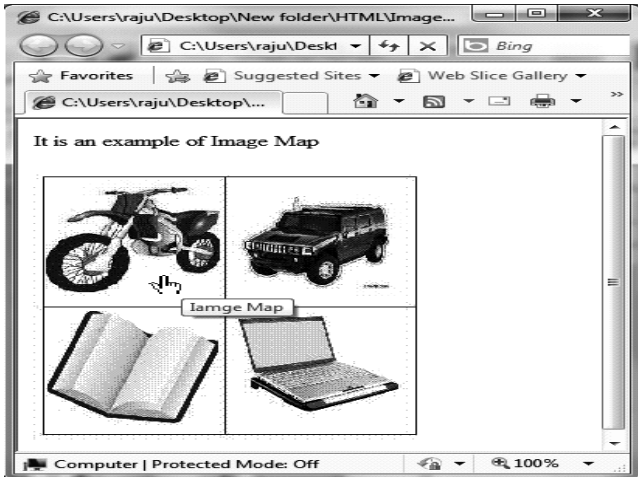


Fig. 6.19

The <***map***> element actually creates the map for the image and usually follows directly after the <img /> element. It acts as a container for the <area> elements that actually define the clickable hotspots. The <map> element carries only one attribute, the name attribute, which is the name that identifies the map. This is how the <img /> element knows which <map> element to use.

The <***area***> element specifies the shape and the coordinates that define the boundaries of each clickable hot-spot.

Here's an example from the image map that was used for the image in Fig.

```
<html>
<body>
<p> It is an example of Image Map </p>
<img src="ImageMap.gif" alt="Iamge Map" border="0" usemap="#gallery"/>
<map name="gallery">
<area shape="rect" coords="5,13,122,157" href="bike.html"
target="_self">
<area shape="rect" coords="122,12,246,157" href="car.html"
target="_self">
<area shape="rect" coords="6,159,121,294" href="book.html"
target="_self">
<area shape="rect" coords="124,157,243,299" nohref="car.html" >
</map>
</body>
</html>
```

Advantages of Image Maps

- Since an image-map is one large image, you can place links in it the way you want without worrying about layout issues.

- Lesser HTML scripting is involved since there are programs available that will generate the code.
- Image map creation is quite simple if you use a program such as MapThis.

Disadvantages of Image Maps

- An image-map takes only one ALT attribute. Thus, it's not possible to have different ALT attributes for different links. Visitors with disability or those using text based browsers might not appreciate this.
- No interactivity is possible. You can't have separate mouseovers for individual links.
- An image map is one large image and if its not optimized properly, will take a long time to download.

6.3 LAYOUT WITH TABLES

Tables are a powerful HTML tool that can be used in many ways. In most applications, a table is commonly used to present tabular information such as schedules, statistics, calendars, etc. In the world of HTML, a table is also used for these purposes, but it is more commonly used for organizing the content of complex web pages. Many web designers prefer to create on overall web site layout through the use of a single table. A table lets web designers make use of rows and columns so they can specify precisely where text, images, and objects are to be placed within their documents. Understand How Tables Work and When to Use Them

Quite simply, a table is a section of information, broken up into columns and/or rows of blocks, called cells. (See Figure 6-1).

- A ***cell*** is the basic unit of a table. It is formed by the intersection of a row and a column.
- A ***row*** is a line of cells running from left to right.
- A ***column*** is a stack of cells running up and down.
- The ***border*** is the rectangle that defines each cell and the outside of the table.

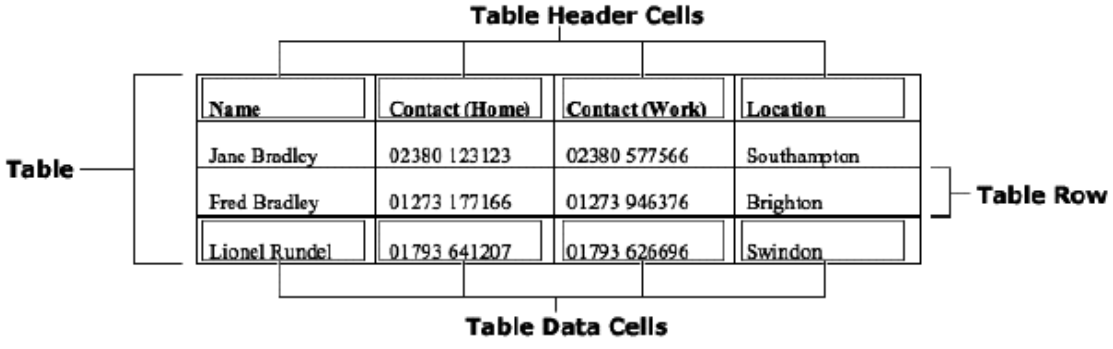


Fig. 6.20

6.3.1 Table's Structure

To create a table, you use the following tags:

1. The table tag is a container for every other tag used to create a table in

HTML. The opening <**table**> and closing </table> tags should be placed at the beginning and the end of your table.

2. The <**tr**> (table row) tag marks the start of each row, and the </tr> tag marks the end of each row. The number of pairs of <tr> and </tr> tags you use controls the number of rows in the table.
3. The <**th**> tag stands for table header. An optional tag used instead of the td tag, this tag defines a cell containing header information. By default, the content in header cells is\ bolded and centered.
4. The <**td**> (table data) tag marks the start of each cell within a row, and then the <**/td**> tag marks the end of each cell. The number of pairs of <td> and </td> tags controls the number of cells in the row, and thus controls the number of columns.

```
<table>
<tr><td> Cell 1</td> <td> Cell 2 </td> </tr>
<tr><td> Cell 1 </td> <td> Cell 2 </td><td> Cell 3 </td></tr>
<tr><td> Cell 1 </td></tr>
</table>
```

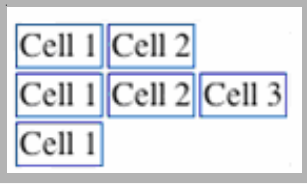


Fig. 6.21

Opening and closing table tags surround the entire section of code. This tells the browser that everything inside these tags belongs in the table. And there are opening and closing **tr** tags for each row in the table. These surround **td or th** tags, which, in turn, contain the actual content to be displayed by the browser.

Format Tables within Web Pages : You may have noticed by now that all the text in a table appears aligned to the left side of each cell. This, and may other features of a table, can be easily customized with a few table attributes or a style sheet.

Borders : Tables, by nature of their design, have internal and external borders. By default, most recent browsers set the border size to zero, making them invisible. However, borders can be quite useful for tables of statistical information, for example, where it's necessary to see the columns to understand the data better. The key is understanding the **three attributes** related to the use of these borders.

The Border Attribute : To set the border width for the outside **border** of a table, add the border attribute to the opening <**table**> tag, and specify the number of pixels for the width of the border. Specifying a border width without a border color makes the browser display the border in its default color, which is usually gray. The best way to set border color is to use CSS. For example, to create a border five pixels wide around the table:

```
<table border="5" style="border-color:black">
```

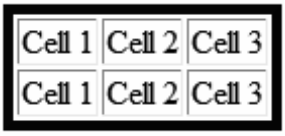


Fig. 6.22

The frame Attribute : To control which outside borders are displayed for a table, add the **frame** attribute to the opening <**table**> tag, and specify one of the values explained in **Table 6-1.**

```
<table border="2" frame="hsides">
```

Remember, to use the `frame` attribute, the `border` attribute must be set to a whole number greater than zero

Cell 1	Cell 2	Cell 3
Cell 1	Cell 2	Cell 3

Fig. 6.23

The rules Attribute : To control which inside borders of a table are displayed, add the **rules** attribute to the opening <**table**> tag, and specify one of the values explained in **Table 6.24**. Rules are the lines or borders around the individual cells. For example, to display only the horizontal borders within a table:

```
<table border="5px" frame="box" rules="none">
```

Cell 1	Cell 2	Cell 3
Cell 1	Cell 2	Cell 3

Fig. 6.24

VALUE	EFFECT
box	Displays the outside borders (same as border)
border	Displays the outside borders (same as box)
void	Hides all the outside borders
above	Displays the top border
below	Displays the bottom border
hsides	Displays the top and bottom borders
lhs	Displays the left border
rhs	Displays the right border
vsides	Displays both left and right borders

Table: Values for the frame Attribute

VALUE	EFFECT
all	Displays rules around all cells
none	Displays no rules
groups	Displays rules around the horizontal or vertical groups that are defined. See "Group Cells by Rows and Columns" later in this chapter for details
rows	Displays all horizontal rules
cols	Displays all vertical rules

Table: Values for the Rule Attribute

6.3.2 Cell Padding and Spacing

When the borders are visible for a table, it's easier to see how much space is around the content and in between the cells. Two attributes can be added to the table tag, so you can control those types of spaces.

1. **Cellpadding** Space between the content within the cell and the edges of that cell.
2. **Cellspacing** Space in between each of the individual cells

First, **CellPadding** affects the amount of space between the content and the edge of the cell. When the borders are visible, increasing the cell padding can give extra buffer space around the text, so it doesn't run into the borders.

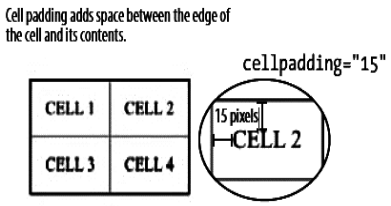


Fig. 6.25

```
<table border="5" cellpadding="15" cellspacing="5">
```

Second, **CellSpacing** affects the amount of space between each of the cells in the table. While not located inside the actual cells, this space can be increased to allow for a gutter between multiple cells.

```
<table border="5" cellpadding="5" cellspacing="15">
```

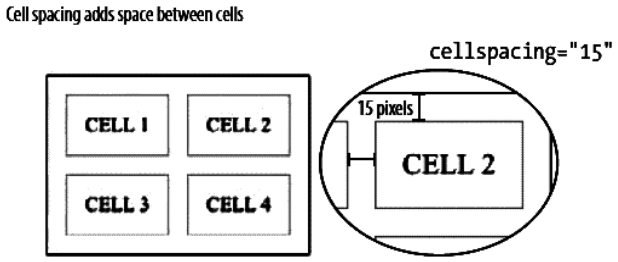


Fig. 6.26

6.3.3 Set Table and Cell Width

Planning out the size of your tables ahead of time, this is particularly important if the table you are creating needs to fit within a predetermined amount of space on your page. You can use the height and width attributes independently of each other. If you don't specify them in your HTML, the browser chooses the size based on the amount of content within each cell and the amount of available space in the window.

However, you can choose to control the exact size of the entire table by using width and/or height styles in the <**table**> tag. You can also control the size of each cell by putting width and height styles in the individual <**td**> tags. The width and height styles can be specified as either pixels or percentages.

```
<table border="3" height="200" width="200">
```

In this case, I would specify an absolute size for my table, one that shouldn't change if the browser window were larger or smaller.

```
<table border="3" height="50%" width="50%">
```

This is called relative sizing because I'm not specifying absolute pixel dimensions but, instead, sizes that are relative to the browser window opening, with relative sizing, the table size varies according to the window size.

<table border="5" `width="600">`

Table with the width attribute of 600 pixels and having the border of 5 pixels.

```
<table width="90%" border="1">
<tr>
<td width="200">This cell is 200 pixels wide.</td>
<td width="140">This cell is 140 pixels wide.</td>
<td>This cell's width is set automatically.</td>
</tr>
</table>
```

This cell is 200 pixels wide.	This cell is 140 pixels wide.	This cell's width is set automatically.

Specify Cell Width

Fig. 6.27

6.3.4 Align a Table, Row, or Cell

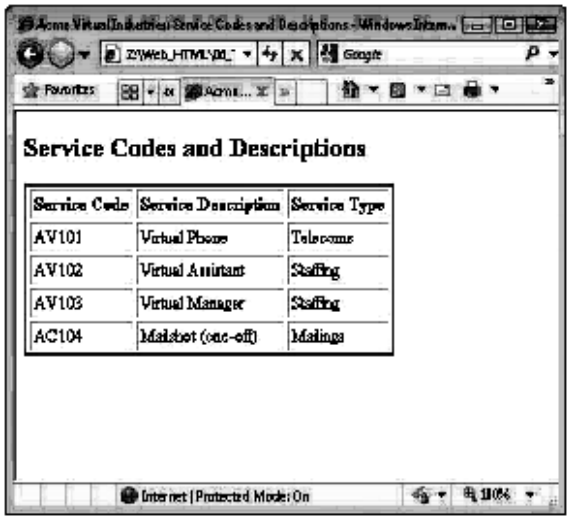
To achieve the placement you want, you can align a whole table, a whole row, or the contents of individual cells.

Align a Table Horizontally : You can align a table horizontally within a web page by adding the align attribute to the opening <table> tag and specifying left, right, or center, as needed. For example, to center a table on the web page that contains it **(see Figure 6-5):**

```
<table align="center" border="2" cellspacing="5">
```

Align a row horizontally : You can align the contents of a row within their cells by adding the align attribute to the opening <tr> tag and specifying left, right, center, or justify, as needed. For example, to apply justified alignment (where both the left and right edges of text are aligned) to a row:

<tr align="justify">



Default Left Alignment

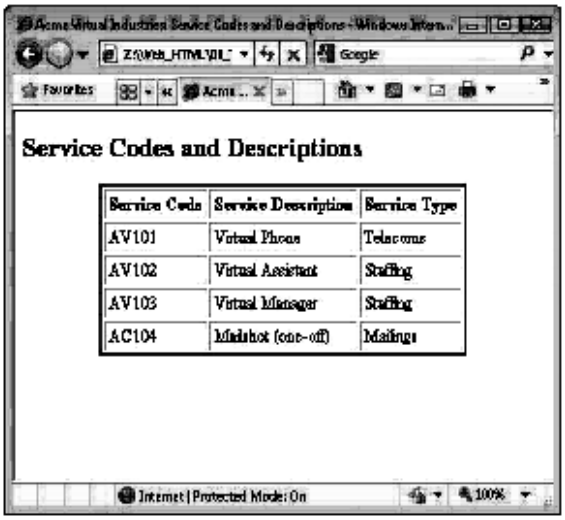


Table Center Align

Fig. 6.28

Align a cell's contents horizontally : You can align a cell's contents horizontally by adding the align attribute to the opening <td> tag and specifying left, right, center, or justify, as needed.

```
<td align="center">Telecoms</td>
```

6.3.5 Cell Spanning

Tables that use a regular grid are useful for presenting data in a tabular format, but to lay out a page with tables, you'll often need to remove some of the borders between rows and columns. To do so, you make a cell span two or more rows or columns-in other words, you merge the cells in two or more rows, or in two or more columns, into a single larger cell.

Make a cell span two columns : To make a cell span two or more columns, add the colspan attribute to the opening <td> tag of the cell in the leftmost of the columns involved, and specify the number of columns to span. For example, to make a cell span three columns:

```
<table border="1" cellspacing="2" cellpadding="4">
<tr> <td colspan="3">This cell spans all three columns.</td></tr>
<tr>
    <td>Column 1</td>
    <td>Column 2</td>
    <td>Column 3</td>
</tr>
</table>
```

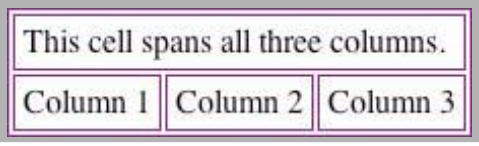


Fig. 6.29

Make a Cell Span two Rows : To make a cell span two or more rows, add the rowspan attribute to the opening <td> tag of the cell in the topmost of the rows involved, and specify the number of rows to span.

For example, to make a cell span two rows:

```
<table border="1">
    <tr>
        <td colspan="2">This cell spans the two columns </td>
        <td rowspan="2">This cell spans two rows</td>
    </tr>
    <tr>
        <td>column 1</td>
        <td> column 2</td>
    </tr>
</table>
```

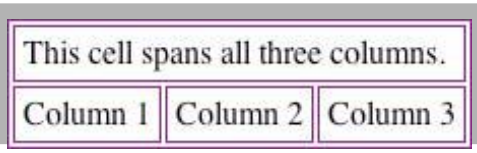


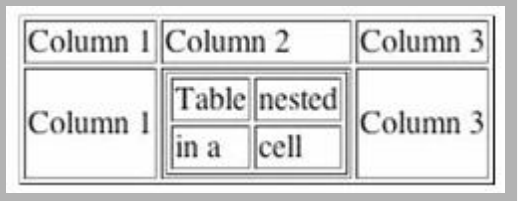
Fig. 6.30

6.3.6 Create a Nested Table

Spanning columns, spanning rows, or spanning both columns and rows gives you decent flexibility in laying out your tables; however, if you must create a truly intricate table design, you may need to nest one table within another.

To nest a table, enter the complete structure of the nested table within the <td> and </td> tags for the cell in which you want the nested table to appear. The next example creates the simple nested table shown here.

```
<table border="1">
    <tr><td>Column 1</td><td>Column 2</td><td>Column 3</td></tr>
    <tr>
        <td>Column 1</td>
        <td>
           <table border="1">
               <tr>
                      <td>Table</td>
                        <td>nested</
td>
                </tr>
                 <tr>
                          <td>in a</
td>
                         <td>cell</td>
                   </tr>
                     </table>
    </td>
    <td>Column 3</td>
     </tr>
</table>
```



Nested Tables

Fig. 6.31

6.3.7 Create a Vertical Line

The <hr> tag lets you easily insert a horizontal line in a web page. To insert a vertical line, create a two-column table and specify frame="void" and rules="cols", as shown in this example:

```
<table frame="void" rules="cols" >
    <tr> <td>Cell 1</td>  <td>Cell 2</td> </tr>
    <tr><td>Cell 3 </td> <td>Cell 4</td>  </tr>
</table>
```

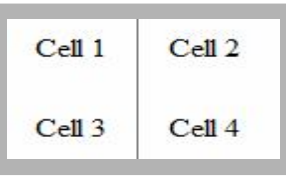


Fig. 6.32

6.3.8 Additional Formatting Techniques

HTML 4.0 introduced some new tags geared toward helping Web developers build more user-friendly tables.

Apply a Background Color or Picture : To make a table more colorful or distinctive, you can apply a background color or a background picture.

Apply a background color to a table, row, or cell : To apply a background color to a table, add the bgcolor attribute to the opening <table> tag, and specify the color either by name or by hexadecimal code. For example, to apply yellow as the background color:

```
<table bgcolor="yellow">
```

To apply a background color to all cells in a row, add the bgcolor attribute to the opening <tr> tag. For example:

```
<tr bgcolor="black">
```

To apply a background color to an individual cell, add the bgcolor attribute to the opening <td> tag. For example:

```
<td bgcolor="blue">
```

Background Images : HTML enables you to add a background image to your table. This is accomplished by adding the background attribute to the table tag, similar to how you use the background attribute in the body tag to add a background image to your entire Web page.

```
<table background="images/clouds.jpg">
```

Captions : The optional caption tag enables you to specify captions for your tables. This isn't an attribute of the table tag; it's a stand-alone element used after the table tag, but before the first table row.

```
<caption align="bottom"><b>This is caption for table.</b></caption>
```

6.3.9 Group Cells by Rows and Columns

The frame and rules attributes of the <table> tag enable you to create many arrangements of borders in your tables. But if you need more flexibility, you can use rules="groups" to put borders only on specific groups of rows and columns.

Create Groups of Rows :

1. To create groups of rows, you split your table into a section of header rows, a body section, and a section of footer rows.
2. To create the header section, put an opening <thead> tag before the header rows and an ending </thead> tag after them.
3. Within the header section, you can use either <th> and </th> tags to create a table header (which is boldface and centered) or <td> and </td> tags to create standard table cells.
4. To create the footer section, put an opening <tfoot> tag before the footer rows and an ending </tfoot> tag after them.

5. To create the body section, put an opening <tbody> tag before the body rows and an ending </tbody> tag after them.

The next example shows a short table divided into header, body, and footer sections:

```
<table rules="groups" width="180" border="4px" >
<thead>
     <tr>
          <td>Member #</td>
          <td align="right">First</td>
          <td>Last</td>
     </tr>
</thead>
<tfoot>
     <tr>
          <td align="right">Member</td>
          <td>count:</td>
          <td>2</td>
     </tr>
</tfoot>
<tbody>
     <tr>
          <td>1007</td> Grouped Rows
          <td align="right">Jack</td>
          <td>Hobbs</td>
</tr>
<tr>
          <td>1008</td>
          <td align="right">Katja</td>
          <td>Reina</td>
</tr>
          </tbody>
          </table>
```

Member #	First Last
1007	Jack Hobbs
1008	Katja Reina
Member count: 2	

Grouped Rows

Fig. 6.33

This HTML code produces the table shown here, with the rules="groups" statement producing borders across the rows in the groups defined.

Create Groups of Columns : To create groups of columns, you split your table by using the <colgroup> tag with the span attribute to specify which columns belong in each group. The following example shows a short table divided into two column groups, each of which contains two columns:

```
<table rules="groups" width="240" border="4px" >
     <colgroup span="2"></colgroup>
     <colgroup span="2"></colgroup>
          <tr>
               <td>Employee</td>
               <td>Item</td>
               <td>Quantity</td>
               <td>Total $</td>
          </tr>
          <tr>
               <td>Johns</td>
               <td>A384</td>
               <td align="right">48</
td>
               <td align="right"> 480.00</td>
          </tr>
          <tr>
               <td>Bills</td>
               <td>C839</td>
               <td align="right">11</td>
               <td align="right">4492.00</td>
          </tr>
          <tr>
               <td>Acinth</td>
               <td>X420</td>
               <td align="right">88</td>
               <td align="right">6295.00</td>
          </tr>
          </table>
```

Employee	Item	Quantity	Total $
Johns	A384	48	480.00
Bills	C839	11	4492.00
Acinth	X420	88	6295.00

Grouped Columns

Fig. 6.34

This HTML code produces the table shown here, with the rules="groups" statement producing a vertical border between the groups of columns and no border between the columns that make up each group.

6.4 HTML FRAMES

Frames divide a browser window into several pieces or panes, each pane containing a separate XHTML/HTML document. One of the key advantages that frames offer is that you can then load and reload single panes without having to reload the entire contents of the browser window. A collection of frames in the browser window is known as a frameset. The window is divided up into frames in a similar pattern to the way tables are organized: into rows and columns. The simplest of framesets might just divide the screen into two rows, while a complex frameset could use several rows and columns.

To create a frameset document, first you need the <**frameset**> element, which is used instead of the <**body**> element. The frameset defines the rows and columns your page is divided into, which in turn specify where each individual frame will go. Each frame is then represented by a <frame> element.

You also need to learn the <**noframes**> element, which provides a message for users whose browsers do not support frames. Now we will discuss these tags in detail one by one.

Creating Frames - The <frameset> Element :

- The <frameset> tag replaces the <body> element in frameset documents.
- The <frameset> tag defines how to divide the window into frames.
- Each frameset defines a set of rows **or** columns. If you define frames by using rows then horizontal frames are created. If you define frames by using columns then vertical frames are created.
- The values of the rows/columns indicate the amount of screen area each row/column will occupy.
- Each frame is indicated by <frame> tag and it defines what HTML document to put into the frame.

Example : Following is the example to create two horizontal frames –

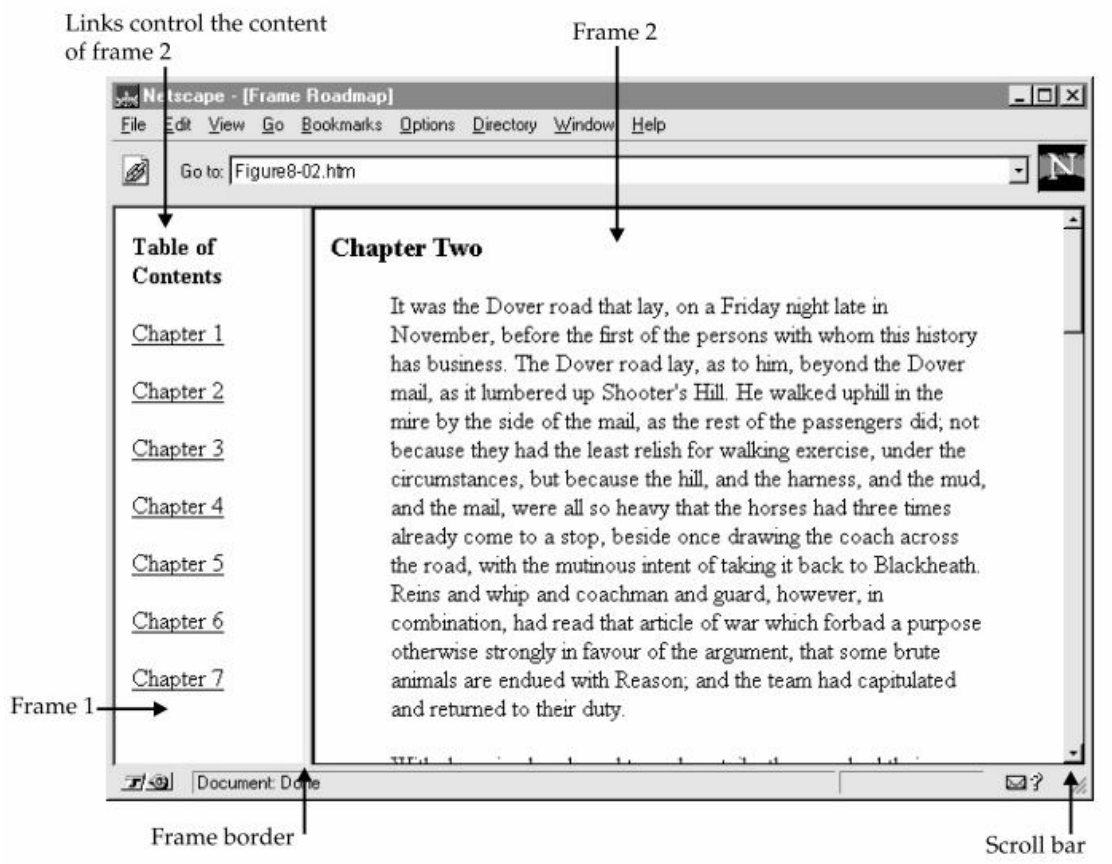


Fig. 6.35

```
<html>
<head>
<title>Frames example</title>
</head>
    <frameset cols="25%,75%">
         <frame src="/html/top_frame.htm" />
         <frame src="/html/main_frame.htm" />
         <noframes>
         <body>
         Your browser does not support frames.
         </body>
         </noframes>
    </frameset>
</html>
```

6.4.1 The <frameset> Element Attributes

Following are important attributes of <frameset> and should be known to you to use frameset.

1. Cols: specifies how many columns are contained in the frameset and the size of each column. You can specify the width of each column in one of four ways:

- Absolute values in pixels. For example to create three vertical frames, use cols="100, 500,100".
- A percentage of the browser window. For example to create three vertical frames, use cols="10%, 80%,10%".
- Using a wildcard symbol. For example to create three vertical frames, use cols="10%, *,10%". In this case wildcard takes remainder of the window.

2. Rows: attribute works just like the cols attribute and can take the same values, but it is used to specify the rows in the frameset. For example to create two horizontal frames, use rows="10%, 90%". You can specify the height of each row in the same way as explained above for columns.

3. Border: attribute specifies the width of the border of each frame in pixels. For example border="5". A value of zero specifies that no border should be there.

6.4.2 The <frame> Element

The <frame> element indicates what goes in each frame of the frameset. The <frame> element is always an empty element, and therefore should not have any content, although each <frame>element should always carry one attribute, src, to indicate the page that should represent that frame.

- **The src Attribute:** The srcattribute indicates the file that should be used in the frame.

```
src="main_page.html"
```

The value for the srcattribute is a normal HTML page, so you must have a corresponding page for each <frame />element.

- **The name Attribute:** The name attribute allows you to give a name to a frame; it is used to indicate which frame a document should be loaded into. This is especially important when you want to create links in one frame that load pages into a second frame, in which case the second frame needs a name to identify itself as the target of the link. You will see more about making links between frames later in the chapter.

```
name="main_frame"
```

The <noframes> Element : If a user's browser does not support frames (which is very rare these days), the contents of the <noframes>element should be displayed to the user. In HTML, you must place a <body>element inside the <noframes>element because the <frameset>element is supposed to replace the <body>element. But if a browser does not understand the <frameset> element, it should ignore these elements and the <noframes> element and understand what is inside the <body>element contained in the <noframes>element.

```
<noframes> <body>This site does not understand frames.</body> </noframes>
```

6.4.3 Creating Links Between Frames

Whenever you use frames on a Web site, you have to deal with the issue of linking between them. By default, whenever a user clicks a link within a frame, the page loads within that same frame. Sometimes it becomes necessary to load a linked page within another frame.

Before you can link to another frame, it must have a name. Earlier in this module, you learned how to use the nameattribute with the frametag to identify a frame by name:

```
<frameset cols="150,*">
<frame src="links.html" name="links" />
<frame src="intro.html" name="intro" />
</frameset>
```

Once the frame has been given a name, you only need to add the target attribute to the link.

Targets : Targets Imagine you have an HTML page for a frameset with two columns, called frameset_cols.html. The frames in this structure are called "links" on the left and "intro" on the right. When a user clicks the words "About Us" in the left frame, you want the "About Us" page to load in the right frame.

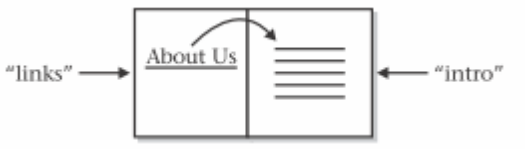


Fig. 6.36

Accomplishing this task requires two steps.

1. Make sure the frame is named (using the name attribute), as in

   <frame src="intro.html" name="intro" />in the frameset HTML document.

2. Then, in the HTML page containing the link, use the target attribute in the a tag to specify in which frame the link should load. For example,

   <a href="page3.html" target="intro">.

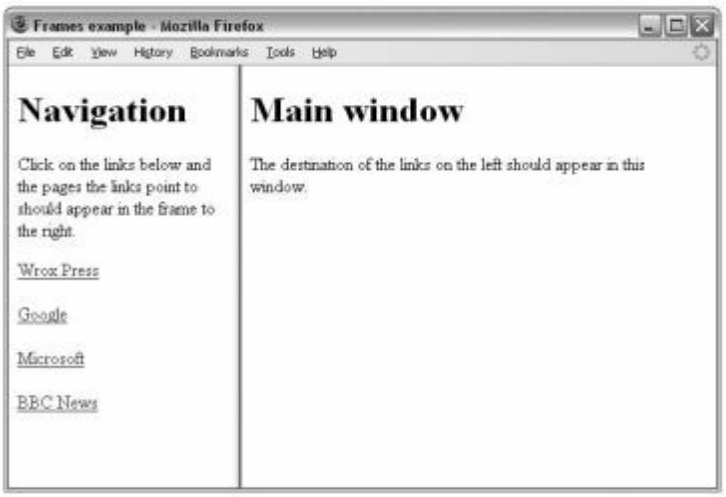


Fig. 6.37

Some particular values for the TARGET attribute have special meaning, which are summarized in Table.

Value	Description
_self	Loads the page into the current frame.
_blank	Loads a page into a new browser window opening a new window.
_parent	Loads the page into the parent window, which in the case of a single frameset is the main browser window.
_top	Loads a page into a new browser window, replacing any current frames.

6.4.4 Nested Framesets

You have seen that a single frameset gives you a fixed grid-like structure of rows and columns just like a table. If you want to create a more complex design, you might choose to use a nested frameset.

You create a nested frameset by using a new <frameset>element in the place of one of the <frame> elements. Take a look at the following example:

```
<frameset rows="*, 300, *">
<frame src="frames/top_frame.html"/>
<frameset cols="*, 400, *">
<frame src="frames/blank.html" />
<frame src="frames/main_frame.html"/>
<frame src="frames/blank.html" />
</frameset>
<frame src="frames/bottom_frame.html"/>
</frameset>
```

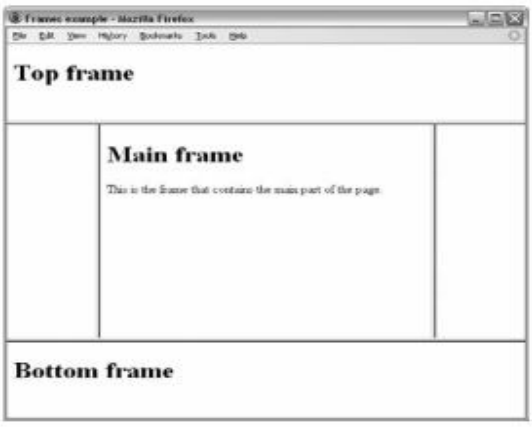


Fig. 6.38

This example creates a set of three rows. In the middle row is a nested frameset with three columns. You can see that the two side columns actually share the same file. Figure shows what this example looks like in a browser.

6.4.5 Inline Frames (Floating Frame)

Up to this point, all the frames shown have been attached to the sides of the browser (left, right, top, or bottom). Another form of frame, called a floating frame (introduced by Microsoft), has been incorporated into the HTML 4 standard. The idea of the floating frame is to create an inline framed region, or window, that acts similarly to any other embedded object, insofar as text can be flowed around it. An inline frame is defined by the <IFRAME> element and may occur anywhere within the <BODY> element of an HTML document. Compare this to the <FRAME> element, which should occur only within the <FRAMESET> element; remember that the <FRAMESET> element should preclude the <BODY> element.

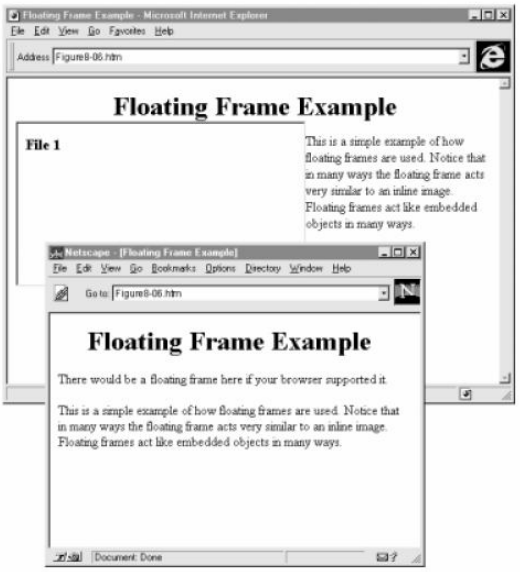


Fig. 6.39

The major attributes to set for the <IFRAME> element include SRC, HEIGHT, and WIDTH. The SRC is set to the URL of the file to load, while the HEIGHT and WIDTH are set to either the pixel or percentage value of the screen that the floating-frame region should consume. Like an <IMG> element, floating frames should support ALIGN, HSPACE, and VSPACE attributes for basic positioning within the flow of text. Note that, unlike the <FRAME> element, the <IFRAME> element comes with a close tag. <IFRAME> and </IFRAME> should contain any HTML markup code and text that is supposed to be displayed in browsers that don't support floating frames.

```
<HTML>
<BODY> <H1 ALIGN="center">Floating Frame Example</H1>
<IFRAME NAME="float1" SRC="fileone.htm" WIDTH="350" HEIGHT="200"
ALIGN="LEFT"> There would be a floating frame here if your browser
supported it.
</IFRAME> <P>This is a simple example of how floating frames are used.
Notice that in many ways the floating frame acts very similar to an
inline image. </P>
</BODY>
</HTML>
```

6.4.6 Layers

The layer function enables the page designer to define precisely positioned, overlapping layers of transparent or opaque content in a page. Besides being able to stack layers on top of each other to create complex layouts, authors can bind the layers to code that can move them around or change the order of overlap. Scripting combined with absolute positioning can truly make pages more dynamic. The basic idea of a layer is similar to the idea of a frame, except that layers may overlap. They generally are defined in the same document, unlike frames, which require multiple documents. A layer defines a region or portion of the browser window that can be manipulated, and may overlap other layers. Layers come in two basic forms:

1. **Positioned layers : Defined by the <LAYER> element**
2. **Inflow layers : Defined by the <ILAYER> element**

Positioned Layers : To define a section of a document as a layer to position, enclose it within <LAYER> and </LAYER>. The layer should be named, just as a frame is named, so that it can be manipulated later. To name a layer, set the ID attribute of the element to a unique identifier. For a positioned layer, specify the upper corner of the layer by setting the TOP and LEFT attributes to the pixel coordinates of the upper-left corner of the layer, relative to the browser window. You may also want to set the WIDTH and HEIGHT attributes of the layer. These take pixel values, so the actual size of the layer can be controlled, regardless of its content. A simple example showing how absolutely positioned layers work is presented here:

```
<LAYER ID="Layer1" TOP="100" LEFT="50" BGCOLOR="lightgreen">
<H2>This is a layer.</H2>
<UL>
<LI>This layer is positioned at 100,150.
</UL>
</LAYER>
```

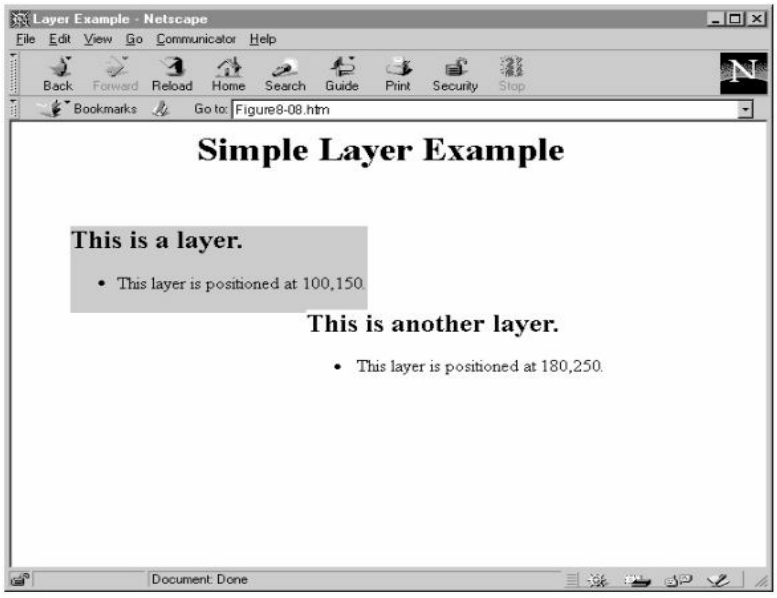


Fig. 6.40

Inflow Layers : Inflow layers are different than positioned layers, because inflow layers fall naturally within the flow of a document, much as an inlined object is positioned, such as an image. To indicate that content is part of an inflow layer, enclose it within <ILAYER> and </ILAYER> tags.

6.5 HTML AND MEDIA TYPES

Multimedia is everything you can hear or see: texts, books, pictures, music, sounds, CDs, videos, DVDs, Records, Films, and more. Multimedia comes in many different formats. On the Internet you will find many of these elements embedded in web pages, and today's web browsers have support for a number of multimedia formats.

Multimedia Formats : Multimedia elements (like sounds or videos) are stored in media files. The most common way to discover the media type is to look at the file extension. When a browser sees the file extensions .htm or .html, it will assume that the file is an HTML page. The .xml extension indicates an XML file, and the .css extension indicates a style sheet.

- Picture formats are recognized by extensions like .gif and .jpg.

Browser Support : The first Internet browsers had support for text only, and even the text support was limited to a single font in a single color, and little or nothing else. Then came web browsers with support for colors, fonts and text styles, and the support for pictures was added.

The support for sounds, animations and videos is handled in different ways by different browsers. Some elements can be handled inline, some requires a plug-in and some requires an ActiveX control.

6.5.1 Audio Support in Browsers

Sound is a vital element of true multimedia Web pages. Sound can be stored in many different formats.

- **The MIDI Format:** The MIDI (Musical Instrument Digital Interface) is a format for sending music information between electronic music devices like synthesizers and PC sound cards. The MIDI format was developed in 1982 by the music industry. The MIDI format is very flexible and can be used for everything from very simple to real professional music making. MIDI files do not contain sampled sound, but a set of digital musical instructions (musical notes) that can be interpreted by your PC's sound card.
- **The RealAudio Format:** The RealAudio format was developed for the Internet by Real Media. The format also supports video. The format allows streaming of audio (on-line music, Internet radio) with low bandwidths. Because of the low bandwidth priority, quality is often reduced. Sounds stored in the RealAudio format have the extension .rm or .ram.
- **The AU Format:** The AU format is supported by many different software systems over a large range of platforms. Sounds stored in the AU format have the extension .au.
- **The AIFF Format:** The AIFF (Audio Interchange File Format) was developed by Apple. AIFF files are not cross-platform and the format is not supported by all web browsers. Sounds stored in the AIFF format have the extension .aif or .aiff.
- **The SND Format:** The SND (Sound) was developed by Apple. SND files are not cross-platform and the format is not supported by all web browsers. Sounds stored in the SND format have the extension .snd.
- **The WAVE Format:** The WAVE (waveform) format is developed by IBM and Microsoft. It is supported by all computers running Windows, and by all the most popular web browsers. Sounds stored in the WAVE format have the extension .wav.
- **The MP3 Format (MPEG):** MP3 files are actually MPEG files. But the MPEG format was originally developed for video by the Moving Pictures Experts Group. We can say that MP3 files are the sound part of the MPEG video format. MP3 is one of the most popular sound formats for music recording. The MP3 encoding system combines good compression (small files) with high quality. Expect all your future software systems to support it. Sounds stored in the MP3 format have the extension .mp3, or .mpga (for MPG Audio).

What Format to Use?

The WAVE format is one of the most popular sound format on the Internet, and it is supported by all popular browsers. If you want recorded sound (music or speech) to be available to all your visitors, you should use the WAVE format. The MP3 format is the new and upcoming format for recorded music. If your website is about recorded music, the MP3 format is the choice of the future.

6.5.2 Video Support in Browser

Like audio files, video files can be compressed to reduce the amount of data being sent. Because of the degree of compression required by video, most video codecs use a lossy approach that involves a trade-off between picture/sound quality and file size, with larger file sizes obviously resulting in longer download times. Video can be stored in many different formats.

- **The AVI Format:** The AVI (Audio Video Interleave) format was developed by Microsoft. The AVI format is supported by all computers running Windows, and by all the most popular web browsers. It is a very common format on the Internet, but not always possible to play on non-Windows computers. Videos stored in the AVI format have the extension.avi.

- **The Windows Media Format:** The Windows Media format is developed by Microsoft. Windows Media is a common format on the Internet, but Windows Media movies cannot be played on non-Windows computer without an extra (free) component installed. Some later Windows Media movies cannot play at all on non-Windows computers because no player is available. Videos stored in the Windows Media format have the extension .wmv.

- **The MPEG Format:** The MPEG (Moving Pictures Expert Group) format is the most popular format on the Internet. It is cross-platform, and supported by all the most popular web browsers.Videos stored in the MPEG format have the extension .mpg or .mpeg.

- **The QuickTime Format:** The QuickTime format is developed by Apple. QuickTime is a common format on the Internet, but QuickTime movies cannot be played on a Windows computer without an extra (free) component installed. Videos stored in the QuickTime format have the extension .mov.

- **The RealVideo Format:** The RealVideo format was developed for the Internet by Real Media. The format allows streaming of video (on-line video, Internet TV) with low bandwidths. Because of the low bandwidth priority, quality is often reduced. Videos stored in the RealVideo format have the extension .rm or .ram.

- **The Shockwave (Flash) Format:** The Shockwave format was developed by Macromedia. The Shockwave format requires an extra component to play. This component comes preinstalled with the latest versions of Netscape and Internet Explorer. Videos stored in the Shockwave format have the extension .swf.

6.5.3 Other Binary Formats in HTML

PDF Format: The term "PDF" stands for "Portable Document Format". The key word is portable, intended to combine the qualities of authenticity, reliability and ease of use together into a single packaged concept.

To be truly portable, an authentic electronic document would have to appear exactly the same way on any computer at any time, at no cost to the user. It will deliver the exact same results in print or on-screen with near-total reliability.

The difference between PDF and formats used for writing (Word, Excel, Power Point, Quark, HTML, etc) is profound. Properly made, PDF files are not subject to the

vagaries of other formats. PDFs are not readily editable - and editing may be explicitly prohibited. A precise snapshot, a PDF file is created at a specific date and time, and in a specific way. You can trust a PDF like you can trust a fax. You can't say that about a Word file!

Adobe Systems invented PDF technology in the early 1990s to smooth the process of moving text and graphics from publishers to printing-presses. At the time, expectations were modest, but no longer. PDF turned out to be the very essence of paper, brought to life in a computer. In creating PDF, Adobe had almost unwittingly invented nothing less than a bridge between the paper and computer worlds.

6.6 STYLE SHEETS

CSS stands for Cascading Style Sheets. It is a way to divide the content from the layout on web pages.

How it works

A style is a definition of fonts, colors, etc.

Each style has a unique name: a selector.

The selectors and their styles are defined in one place.

In your HTML contents you simply refer to the selectors whenever you want to ctivate a certain style.

Advantages

With CSS, you will be able to:

1. Define the look of your pages in one place rather than repeating yourself over and over again throughout your site.
2. Easily change the look of your pages even after they're created. Since the styles are defined in one place you can change the look of the entire site at once.
3. Define font sizes and similar attributes with the same accuracy as you have with a word processor - not being limited to just the seven different font sizes defined in HTML.
4. Position the content of your pages with pixel precision.
5. Redefine entire HTML tags. Say for example, if you wanted the bold tag to be red using a special font - this can be done easily with CSS.
6. Define customized styles for links - such as getting rid of the underline.
7. Define layers that can be positioned on top of each other (often used for menus that pop up).

6.6.1 Positioning with Style Sheets

Absolute Positioning – If you position an element (an image, a table, or whatever) absolutely on your page, it will appear at the exact pixel you specify. Say I wanted a graphic to appear 46 pixels from the top of the page and 80 pixels in from the right, I could do it. The CSS code you'll need to add into the image is

```
img {position: absolute; top: 46px; right: 80px; }
```

You just add in which method of positioning you're using at the start, and then push the image out from the sides it's going to be closest to. You can add the CSS directly into the tag using the style attribute (as shown in the introduction to stylesheets), or you can use classes and ids and put them into your stylesheet. It works the same way. The recommended method is to add classes for layout elements that will appear on every page, but put the code inline for once-off things.

Relative Positioning : An element whose position property has the value relative is first laid out just like a static element. The rendered box is then shifted vertically (according to the top or bottom property) and/or horizontally (according to the left or right property). The properties top, right, bottom, and left can be used to specify by how much the rendered box will be shifted. A positive value means the box will be shifted away from that position, towards the opposite side. For instance, a left value of 20px shifts the box 20 pixels to the right of its original position. Applying a negative value to the opposite side will achieve the same effect: a right value of -20px will accomplish the same result as a left value of 20px. The initial value for these properties is auto, which makes the computed value 0 (zero)-that is, no shift occurs.

Fixed Positioning : Fixed positioning is a subcategory of absolute positioning. An element whose position property is set to fixed always has the viewport as its containing block. For continuous media, such as a computer screen, a fixed element won't move when the document is scrolled. For paged media, a fixed element will be repeated on every page.

Floating : A floated element is one whose float property has a value other than none. The element can be shifted to the left (using the value left) or to the right (using the value right); non-floated content will flow along the side opposite the specified float direction.

6.6.2 Implementing CSS

There are three ways of implement a style sheet:

- External style sheet
- Internal style sheet
- Inline style

CSS Syntax

The CSS syntax consists of a set of rules. These rules have 3 parts:

- selector
- property
- value.

Syntax:

```
selector { property: value }
```

The selector is often the HTML element that you want to style. For example:

```
h1 { color: blue }
```

This code tells the browser to render all occurrences of the HTML *h1* element in blue.

Grouping Selectors : You can apply a style to many selectors if you like. Just separate the selectors with a comma.

h1, h2, h3, h4, h5, h6 { color: blue }

Applying Multiple Properties

To apply more than one property separate each declaration with a semi-colon.

h1 { color:blue; font-family:arial,helvetica,"sans serif" }

You can make your CSS code more readable by spreading your style declarations across multiple lines. You can also indent your code if you like. This doesn't affect how your code is rendered - it just makes it easier for you to read.

```
h1 {
color:blue;
font-family:arial, helvetica,"sans serif";
font-size:150%;
}
```

CSS Class Syntax

In CSS, classes allow you to apply a style to a given *class* of an element. To do this, you link the element to the style by declaring a style for the class, then assigning that class to the element.

You declare a CSS class by using a dot (.) followed by the class name. You make up the class name yourself. After the class name you simply enter the properties/ values that you want to assign to your class.

```
.class-name { property:value; }
```

If you want to use the same class name for multiple elements, but each with a different style, you can prefix the dot with the HTML element name.

```
html-element-name . class-name { property:value; }
```

CSS Class Example

```
<head>
<style type="text/css">
h1.css-section { color:#000099 }
p.css-section { color:#999999; }
</style>
</head>
<body>
<h1 class="css-section">CSS Class</h1>
<p class="css-section">CSS classes can be very useful</p>
</body>
```

This results in: **CSS Class**: CSS classes can be very useful.

CSS ID : IDs allow you to assign a unique identifier to an HTML element. This allows you to define a style that can only be used by the element you assign the ID to.

Syntax : The syntax for declaring a CSS ID is the same as for classes - except that instead of using a dot, you use a hash (#).

```
#id-name { property:value; }
```

Again, similar to classes, if you want to use the same id name for multiple elements, but each with a different style, you can prefix the hash with the HTML element name.

```
html-element-name # id-name { property:value; }
```

CSS ID Example

```
<head>
<style type="text/css">
h1#css-section { color:#000099 }
p#css-section { color:#999999; }
</style>
</head>
<body>
<h1 id="css-section">CSS ID</h1>
<p id="css-section">CSS IDs can be very useful</p>
</body>
```

This results in: **CSS ID**: CSS IDs can be very useful

Styles can be specified:

- in an external CSS file
- inside the head section of an HTML page
- inside an HTML element

6.6.2.1 External Style Sheet

An external style sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing one file. Each page must link to the style sheet using the <link> tag. The <link> tag goes inside the head section:

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

An external style sheet can be written in any text editor. The file should not contain any html tags. Your style sheet should be saved with a .css extension. An example of a style sheet file is shown below:

```
hr {color:sienna;}
p {margin-left:20px;}
body {background-image:url("images/back40.gif");}
```

6.6.2.2 Internal Style Sheet

An internal style sheet should be used when a single document has a unique style. You define internal styles in the head section of an HTML page, by using the <style> tag, like this:

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

6.6.2.3 Inline Styles

An inline style loses many of the advantages of style sheets by mixing content with presentation. Use this method sparingly!

To use inline styles you use the style attribute in the relevant tag. The style attribute can contain any CSS property. The example shows how to change the color and the left margin of a paragraph:

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

6.6.3 Advantages of CSS

- **CSS saves time :** When most of us first learn HTML, we get taught to set the font face, size, colour, style etc every time it occurs on a page. This means we find ourselves typing (or copying & pasting) the same thing over and over again. With CSS, you only have to specify these details once for any element. CSS will automatically apply the specified styles whenever that element occurs.
- **Pages load faster :** Less code means faster download times.
- **Easy maintenance :** To change the style of an element, you only have to make an edit in one place.
- **Superior styles to HTML :** CSS has a much wider array of attributes than HTML.

6.6.4 Disadvantages of CSS

- **Browser compatibility :** Browsers have varying levels of compliance with Style Sheets. This means that some Style Sheet features are supported and some aren't. To confuse things more, some browser manufacturers decide to come up with their own proprietary tags.

Fortunately, browser compatibility is becoming less of an issue as the latest browser versions are much more standards-compliant than their earlier counterparts.

6.7 WORKING WITH FORMS

A web form is a web page that contains fields, areas in which a visitor can enter information or select from a set of predefined options. When a visitor accesses the form, the browser displays the web page as usual, including the form's fields. The visitor can interact with the fields-for example, by typing text into a text field, by selecting a check box or one option button in a group of option buttons, or by choosing an item from a drop-down list. When finished with the form, the visitor clicks a command button that submits the form, usually by running a Common Gateway Interface, or CGI, script written in a programming language such as Perl, C, C++, ASP, PHP, etc.

The form Element Forms : are added to web pages using (no surprise here) the **form** element. The **form** element is a container for all the content of the form, including some number of form controls, such as text entry fields and buttons. It may also contain block elements, (**h1**, **p**, and lists), however, it may not contain another form element.

This sample source document contains a form similar to the one shown in table 1.

```
<html>
<body>
<form action="/cgi-bin/mailinglist.pl" method="post">
<ol>
<li><label for="name">Name:</label>
       <input type="text" name="name" id="name" /></li>
<li><label for="name">Email:</label>
     <input type="text" name="email" id="email" /></li>
</ol>
<input type="submit" value="Submit" />
</fieldset>
</form>
</body>
</html>
```

The output of above code that shows in the browser is

Fig. 6.41

In addition to being a container for form control elements, the **form** element has some attributes that are necessary for interacting with the form-processing program on the server. Let's take a look at each.

The **action** attribute provides the location (URL) of the application or script (sometimes called the action page) that will be used to process the form. The **ACTION** value is required for working forms, though on some browsers a value similar to the base URL of the document will be assumed if the **ACTION** is left out. The **method** attribute specifies how the information should be sent to the server. There are only two methods for sending this encoded data to the server: POST or GET indicated using the **method** attribute in the **form** element.

- **The POST method:** When the form's method is set to POST, the browser sends a separate server request containing some special headers followed by the data. Only the server sees the content of this request, thus it is the best method for sending secure information such as credit card or other personal information.
- **The GET method:** With the GET method, the encoded form data gets tacked right onto the URL sent to the server. GET is not appropriate for forms with private personal or financial information.

Add Input Controls to the Form : Let's discuss some input controls, or ways for users to enter data. For example, for you to enter your name on a form, there must be a space for you to do so. This space is what I'm referring to when I use the phrase input control or simply "control." Figure 2 contains some input controls that are labeled for you.

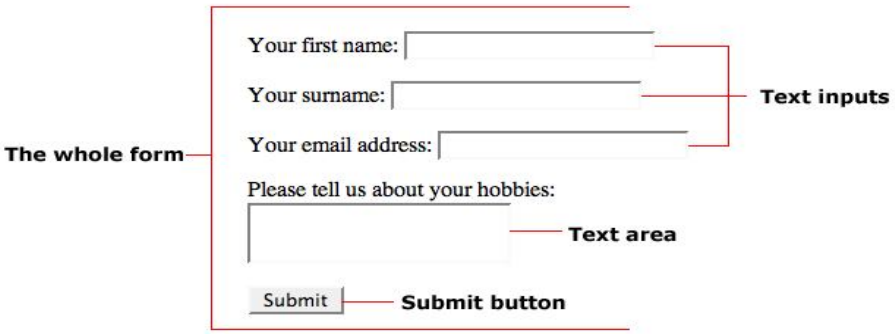


Fig.6.42: An example of a basic input controls.

The majority of these controls are created with an input tag, the actual description of which control you want to include is made through the type attribute, as in the following example:

```
<form>
<input type="text" />
</form>
```

The next sections explain the specific types of controls created with the input tag, as well as a few others that aren't created by it.

6.7.2 Text Box Controls

There are three basic types of text box fields in web forms: single-line text boxes, password boxes, and multiline text entry boxes.

1. **Single-line text box:** One of the simplest types of form control is the text box field used for entering a single word or line of text. It is added to the form using the **input** element with its **type** attribute set to **text**, and the **name** attribute is required for identifying the variable name as shown here and Figure 3.

2. **Password text box:** A password box field works just like a text box field, except the characters are obscured from view using asterisk (*) or bullet (•) characters, or another character determined by the browser.

3. **Multiline text entry box:** When you want your users to be able enter more than just one line of text, for these instances, use the **textarea** element that is replaced by a multi-line, scrollable text entry box when displayed by the browser (Figure 3). Unlike the empty **input** element, the **textarea** element has content between its opening and closing tags. The content of the **textarea** element is the intial content of the text box when the form is displayed in the browser.

```
<form>
<input type="text" name="First  Name"
value="Enter your Name" />
<input type="password" name="pswd" />
<textarea name="contest_entry"
rows="5" cols="100" >
    Tell us why you love the band
    in 50 words or less.
    Five winners will
    get backstage passes!
</textarea>
</form>
```

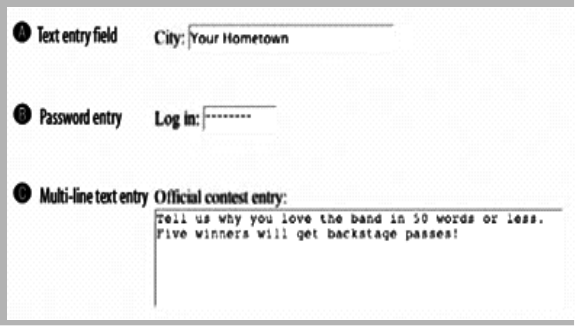


Fig. 6.43

6.7.3 Working with Radio and Checkbox Buttons

Both checkbox and radio buttons make it simple for your visitors to choose from a number of provided options. They are similar in that they function like little on/off switches that can be toggled by the user and are added using the **input** element. They serve distinct functions, however. A form control made up of a collection of radio buttons is appropriate when only one option from the group is permitted, or, in other words, when the selections are mutually exclusive (such as Yes or No, or Male or Female).

When one radio button is "on," all of the others must be "off," sort of the way buttons used to work on old radios-press one button in and the rest pop out.

When checkboxes are grouped together, however, it is possible to select as many or as few from the group as desired. This makes them the right choice for lists in which more than one selection is okay.

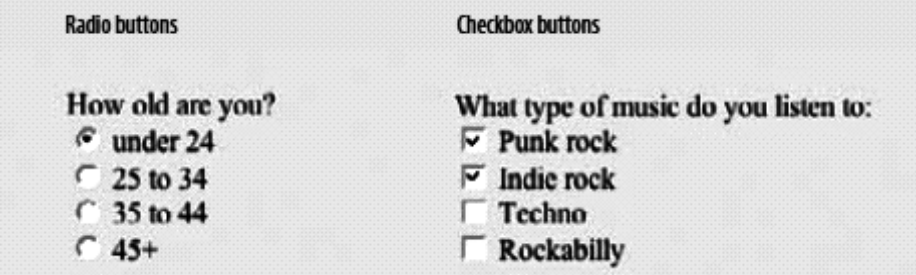


Fig. 4.44 : Radio and Checkbox Input Controls.

6.7.4 Radio Buttons

Radio buttons are added to a form with the **input** element with the **type** attribute set to **radio**. The **name** attribute is required. Here is the syntax for a minimal radio button: In this example, radio buttons are used as an interface for users to enter their age group (a person can't belong to more than one age group, so radio buttons are the right choice). Figure 4 shows how radio buttons are rendered in the browser.

```
<form>
<input type="radio" name="age" value="under24" checked="checked"/> under 24
<input type="radio" name="age" value="25-34" /> 25 to 34
<input type="radio" name="age" value="35-44" /> 35 to 44
<input type="radio" name="age" value="over45" /> 45+
</form>
```

Notice that all of the **input** elements have the same variable name ("age"), but their values are different. Because these are radio buttons, only one button can be checked at a time, and therefore, only one value will be sent to the server for processing when the form is submitted.

You can decide which button is checked when the form loads by adding the **checked** attribute to the **input** element. In this example, the button next to "under 24" will be checked by default.

6.7.5 Checkbox buttons

Checkboxes are added using the **input** element with its **type** set to **checkbox**. As with radio buttons, you create groups of checkboxes by assigning them the same **name** value. The difference, as we've already noted, is that more than one checkbox may be checked at a time. The value of every checked button will be sent to the server when the form is submitted. Here is an example of a group of checkbox buttons used to indicate musical interests. Figure 4 shows how they look in the browser:

```
<form>
<input type="checkbox" name="genre" value="punk" checked="checked"/> Punk rock
<input type="checkbox" name="genre" value="indie" checked="checked" /> Indie rock
<input type="checkbox" name="genre" value="techno" /> Techno
<input type="checkbox" name="genre" value="rockabilly" />Rockabilly
</form>
```

6.7.6 Submit and reset buttons

There are a number of different kinds of buttons that can be added to web forms. The most fundamental is the submit button. When clicked, the submit button immediately sends the collected form data to the server for processing. The reset button returns the form controls to the state they were in when the form loaded.

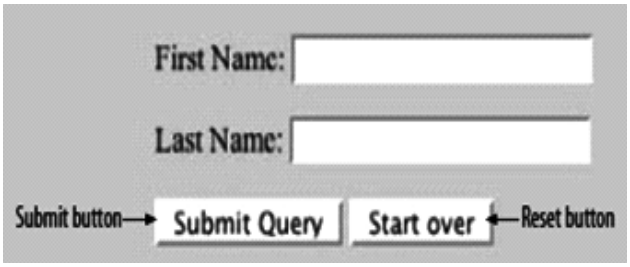


Fig. 4.45

Both submit and reset buttons are added using the **input** element. As mentioned earlier, because these buttons have specific functions that do not include the entry of data, they are the only form control elements that do not require the **name** attribute.

```
<input type="submit" />
<input type="reset" />
```

Submit and reset buttons are straightforward to use. Just place them in the appropriate place in the form, in most cases, at the very end. By default, the submit button displays with the label "Submit" or "Submit Query" and the reset button is labeled "Reset." Change the text on the button using the **value** attribute as shown in the reset button in this example (Figure 5).

```
<p><input type="submit" /> <input type="reset" value="Start over" /></p>
```

6.7.7 Menus

Option for providing a list of choices is to put them in a pull-down or scrolling menu. Menus tend to be more compact than groups of buttons and checkboxes. You add both pull-down and scrolling menus to a form with the select element. Whether the menu pulls down or scrolls is the result of how you specify its size and whether you allow more than one option to be selected.

Let's take a look at both menu types.

6.7.7.1 Pull-down Menus

The select element displays as a pull-down menu and it's a container for a number

of option elements. The content of the chosen option element is what gets passed to the web application when the form is submitted. In pull-down menus, only one item may be selected. Here's an example

```
<select name="EightiesFave" id="form-fave">
<option>The Cure</option>
<option>Cocteau Twins</option>
<option>Tears for Fears</option>
<option>Thompson Twins</option>
<option>Everything But the Girl</option>
<option>Depeche Mode</option>
<option>The Smiths</option>
<option>New Order</option>
</select>
```

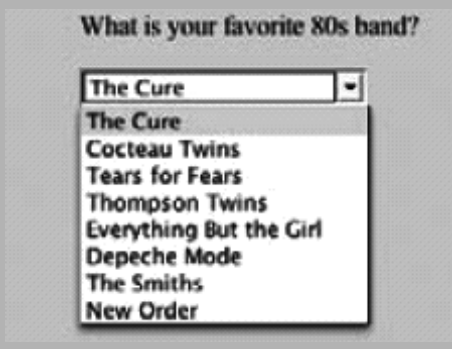


Fig. 4.46

6.7.7.2 Scrolling Menus

To make the menu display as a scrolling list, simply specify the number of lines you'd like to be visible using the size attribute. This example menu has the same options as the previous one, except it has been set to display as a scrolling list that is six lines tall. The multiple attribute allows users to make more than one selection from the scrolling list. Use the selected attribute in an option element to make it the default value for the menu control.

```
<select name="EightiesBands" size="6"
multiple="multiple" for="EightiesBands">
<option>The Cure</option>
<option>Cocteau Twins</option>
<option selected="selected">
Tears for Fears</option>
<option selected="selected">
Thompson Twins</option>
<option>Everything But the Girl</option>
<option>Depeche Mode</option>
</select>
```

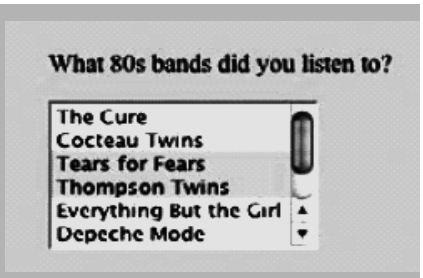


Fig. 4.47

SHORT ANSWER TYPE QUESTIONS

Q.4 What is DHTML? Does working of DHTML depend on the type of browser we use? Explain.

Ans. DHTML stands for Dynamic HTML. DHTML is the combination of HTML and JavaScript. DHTML is the art of combining HTML, JavaScript, DOM, and CSS. According to the World Wide Web Consortium (W3C): "Dynamic HTML is a term used by some vendors to describe the combination of HTML, style sheets and scripts that allows documents to be animated."

DHTML is combination of several built-in browser features in fourth generation browsers that enable a web page to be more dynamic.

Q.9 How do we create tables in HTML? Explain with suitable example.

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

Representing a table involves several kinds of HTML tags:

1. TABLE tags, which surround the entire table specification
2. an optional CAPTION element specifying the caption (name) of the table
3. TR tags, which specify the table rows
4. TH tags, which specify table row and column headers
5. TD tags, which specify the data in the table, i.e. the contents of table cells.

Q.10 How to define an image in html document?

Ans. HTML Images - The <img> Tag and the Src Attribute In HTML, images are defined with the <img> tag. The <img> tag is empty, which means that it contains attributes only, and has no closing tag.

Attributes of the IMG element :

1. **ALIGN :** Alignment of the image with respect to the text baseline.
2. **ALT :** Text to use in place of the referenced image resource, for example due to processing constraints or user preference.
3. **SRC :** Specifies the URI of the image resource. The URL points to the location where the image is stored.

Syntax for defining an image:

```
<IMG SRC="triangle.xbm" ALT="Warning:" /> Be sure to read these instructions.
```

Q.11 How Hyperlinks are created in HTML?

Ans. The LINK element represents a hyperlink. Any number of LINK elements may occur in the HEAD element of an HTML document. It has the same attributes as the <A> element. The LINK element is typically used to indicate authorship,

related indexes and glossaries, older or more recent versions, document hierarchy, associated resources such as style sheets, etc.

```
<a href="http://machine/htbin/imagemap/sample"> HTML Image Map </a>
```

Q.13 How forms are created in HTML?

Ans. A form is a template for a form data set and an associated method and action L. A form data set is a sequence of name/value pair fields. The names are specified on the NAME attributes of form input elements, and the values are given initial values by various forms of markup and edited by the user. The resulting form data set is used to access an information service as a function of the action and method.

Form Element: The FORM element contains a sequence of input elements, along with document structuring elements. The attributes are :

1. **ACTION :** Specifies the action URL for the form. The action URL of a form defaults to the base URI of the document.
2. **METHOD :** Selects a method of accessing the action L. The set of applicable methods is a function of the scheme of the action L of the form.
3. **ENCTYPE :** Specifies the media type used to encode the name/value pairs for transport, in case the protocol does not itself impose a format.

```
<form action="/cgi-bin/mailinglist.pl" method="post">
```

Q.14 Describe any four input types you can use in a form on the web page.

Ans. An HTML form is a document containing text content, markup, special elements called controls (checkboxes, radio buttons, menus, etc.), and labels on those controls.

The control type defined by the INPUT element depends on the value of the type attribute:

Text: To Creates a single-line text input control:-

```
<input type="text" name="textbox1" value="" />
```

Buttons: Authors may create three types of buttons: submit buttons, reset button and push buttons: Authors should specify the scripting language of a push button script through a default script declaration . Authors create buttons with the BUTTON element or the INPUT element.

```
<input type="submit" value="OK" />
```

Checkboxes: Checkboxes are on/off switches that may be toggled by the user. A switch is "on" when the control element's checked attribute is set. When a form is submitted, only "on" checkbox controls can become successful. Several checkboxes in a form may share the same control name. Thus, for example, checkboxes allow users to select several values for the same property. The INPUT element is used to create a checkbox control.

```
<input type="checkbox" name="Chechbox1" value="English" checked="checked" />
```

Radio buttons: Radio buttons are like checkboxes except that when several share the same control name, they are mutually exclusive: when one is switched "on", all others with the same name are switched "off". The INPUT element is used to create a radio button control.

```
<input type="radio" name="sex" value="male" checked="checked" />
```

Q.16 Explain frames in HTML.

Ans. HTML Frames with frames, we can display more than one HTML document in the same browser window. Each HTML document is called a frame, and each frame is independent of the others. The disadvantages of using frames are:

1. Frames are not expected to be supported in future versions of HTML
2. Frames are difficult to use. (Printing the entire page is difficult).
3. The web developer must keep track of more HTML documents

The HTML frameset Element: The frameset element holds one or more frame elements. Each frame element can hold a separate document. The frameset element states HOW MANY columns or rows there will be in the frameset, and HOW MUCH percentage/ pixels of space will occupy each of them.

The HTML frame Element: The <frame> tag defines one particular window (frame) within a frameset.

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

Q.17 What is an Image Map ? Create a Image Map.

Ans. Image maps allow you to specify several links that correspond to different areas of one single image, so that when users click different parts of the image they get taken to different pages.

```
<html>
<body>
<p>Click on the sun or on one of the planets to watch it closer:</p>
<img src="planets.gif" width="145" height="126" alt="Planets"
usemap="#planetmap" />
<map name="planetmap">
 <area shape="rect" coords="0,0,82,126" alt="Sun" href="sun.htm" />
 <area shape="circle" coords="90,58,3" alt="Mercury" href="mercur.htm" />
 <area shape="circle" coords="124,58,8" alt="Venus" href="venus.htm" />
</map>
</body>
</html>
```

Q.18 What is HTML style sheets ?

Ans. Web browsers have always had a standard set of styles such as paragraphs are divided by <P> tags and for heading tags <H1> through <H6> tags are used.. Styles enable Web authors to set the styles used for these standard tags, and create new tags with new formatting. A style is a set of formatting commands that dictate how text is formatted. Instead of adding a series of separate commands to text, you apply one style that contains all the commands. In HTML, a style sets specific formatting characteristics for tags on your page.. HTML 4 supports many types of style sheets, including cascading style sheets (CSS), W3C's proposed way to format text on Web pages, which allows various levels of style sheets to work together. A particular Web page might contain more definitions of styles used only on that page.

Q.19 What is CSS?

Ans. CSS stands for Cascading Style Sheets and is used to achieve a level of uniformity easily throughout your web pages. You can a set of styles to as many html files as you want, allowing you to control the style of many web pages all at once1. For example, we can specify that all the background colors be mauve, or that all text be written in Arial font. CSS is written in a language a little different from HTML.

Q.20 How CSS can be implemented ?

Ans. CSS can be implemented the CSS in the following ways :

1. Inline styles 2. Internal styles 3. External styles

Inline : Inline uses of CSS is generally not recommended and is slowly being faded out. Inline CSS is where you stick the style directly inside a HTML tag. For example:

```
<P STYLE="color:green">
The text in this paragraph would then be green.
</P>
```

Internal/Embedded: You can include the page specific CSS code within the <HEAD>

section of your page. While other styles of your external style sheet come through, the background color style of the external sheet will be overridden by the internal stylesheet in the page. Now the CSS code needs to be wrapped with special <STYLE> tags in the HTML:

```
<head>
<style type="text/css">
<!--BODY { background-color: blue;}-->
</style>
</head>
```

External: This is the most common implementation of styles. The CSS code is in a

separate file with a ".css" extension (NOTE: You must have the .css extension). A snippet is put in the <HEAD> section of the HTML file specifying where the style sheet is. For example, if you had a style sheet called main.css in your styles folder under your web directory, your HTML will be:

```
<HEAD>
<LINK REL="stylesheet" TYPE="text/css" HREF="./styles/main.css" />
</HEAD>
```

MULTIPLE CHOICE QUESTIONS

1. What is the difference between XML and HTML?

a) HTML is used for exchanging data, XML is not.

b) XML is used for exchanging data, HTML is not.

c) HTML can have user defined tags, XML cannot

d) Both b and c above

2. What is the attribute for <image> tag?

a) pt b) url

c) path d) src

3. Can a data cell cont an images?

a) Yes b) No

4. Which of the following tags below are used for a multi-line text input control?

a) textml tag b) text tag

c) textarea tag d) Both b and c above

5. Frames are used to divide the browser window into:

a) Dependent segments b) Two dependent segments

c) Four independent segments d) Various independent segments

6. <SCRIPT> ... </SCRIPT> tag can be placed within _____

a) Header b) Body

c) both A and B d) none of the above

7. <TD> ... </TD> tag is used for _____

a) Table heading b) Table Records

c) Table row d) none of the above

8. Which is true to change the text color to red?

a) <BODY BGCOLOR=RED> b) <BODY TEXT=RED>

c) <BODY COLOR=RED> d) none of the above

9. What is the correct syntax in HTML for creating a link on a webpage?

a) <LINK SRC= "ghumti.html"> b) <BODY LINK = "ghumti.html">

c) <A SRC = "ghumti.html" > d) < A HREF = "ghumti.html">

10. Which of the following is an attribute of <Table> tag?

a) SRC b) LINK

c) CELLPADDING d) BOLD

11. How to define target in new page in HTML?

a) Click Here

b) Click Here

c) Click Here

d) Click Here

12. What is <tt> tag in HTML?

a) It renders fonts as teletype text font style.

b) It renders fonts as truetype text font style.

c) It renders fonts as truncate text font style.

d) None of the Above.

13. What is the use of Forms in HTML?

a) to display contents of email. b) to display animation effect.

c) to collect user's input. d) None of the Above.

14. What is the use of iframe in HTML?

a) to display a web page within a web page.

b) to display a web page with animation effect.

c) to display a web page without browser.

d) All of the Above.

KEY TO OBJECTIVE QUESTIONS

1.a	2.d	3.a	4.c	5.d	6.c
7.b	8.b	9.d	10.c	11.b	12.a
13.c	14.a				

IMPORTANT QUESTIONS

Q.1 What are purpose and syntax of the following HTML tags ? Explain with examples:

(i) Anchor Tag (ii) Unordered Tag.

(iii) Semantic Tag. (iv) Image Tag.

(v) Form Tag.

Q.2 How would you create a table? Describe the process of (1) dividing the table into columns (2) giving header to the table columns (3) adding a border to a table (4) changing the color of a cell.

Q.3 Explain in brief Audio support in Browser.

Q.4 What do you understand by an image map? What is the use of it? Explain.

Q.5 Explain the procedure of creating a rectangular and a circular hotspot in an image map using suitable examples.

Q.6 What do you mean by Cascading Style Sheets ? What are different techniques to embedding CSS into HTML pages ? Explain with example.

Q.7 Write short note on Frames &b Layers with examples.

Q.8 How Cell Padding is differ from Cell Spacing?

Q.9 How to link the Images with in HTML? How we define Src Attribute and Alt Attribute in HTML?

Q.10 How you understand about Forms output? How you use Form's Action Attribute and Submit Button in HTML?

Q.11 What are the Hyperlink,Anchors and Link in the HTML?

Q.12 What are Style Sheets? What are the style precedence rules when using multiple approaches?

****

COMMON GATEWAY INTERFACE

IN THIS CHAPTER, YOU WILL LEARN

- Introduction to CGI
- The Hypertext Transport protocol
- Browser requests, Server Responses
- Proxies, Content Negotiation
- The CGI Environment, Environment variables,
- forms and CGI, CGI Output, Sending Data to the server
- Architectural Guidelines, Coding Guidelines
- Efficiency and optimization.

7.1 INTRODUCTION TO CGI

CGI stands for "Common Gateway Interface". CGI is one method by which a web server can obtain data from (or send data to) databases, documents, and other programs, and presents that data to viewers via the web. More simply, a CGI is a program intended to be run on the web. A CGI program can be written in any programming language, but Perl is one of the most popular, and other languages are: If you're going to create web pages, then at some point you'll want to add a counter, a form to let visitors send you mail or place an order, or something similar. CGI enables you to do that and much more. From mail-forms and counter programs, to the most complex database programs that generate entire websites on-the-fly, CGI programs deliver a broad spectrum of content on the web today.When a web server gets a request for a static web page, the web server finds the corresponding HTML file on its filesystem. When a web server gets a request for a CGI script, the web server executes the CGI script as another process (i.e., a separate application); the server passes this process some parameters and collects its output, which it then returns to the client just as if had been fetched from a static file.

CGI programming involves designing and writing programs that receive their starting commands from a Web page-usually, a Web page that uses an HTML form to initiate the CGI program. The HTML form has become the method of choice for sending data across the Net because of the ease of setting up a user interface using the HTML Form and Input tags. With the HTML form, you can set up input windows, pull-down menus, checkboxes, radio buttons, and more with very little

effort. In addition, the data from all these data-entry methods is formatted automatically and sent for you when you use the HTML form.

Before your CGI program is initiated, the WWW server already has created a special processing environment for your CGI program in which to operate. That environment includes translating all the incoming HTTP request headers into environment variables that your CGI program can use for all kinds of valuable information. In addition to system information (such as the current date), the environment includes information about who is calling your CGI program, from where your program is being called, and possibly even state information to help you keep track of a single Web visitor's actions. State information is anything that keeps track of what your program did the last time it was called.

Why is it called gateway?

Your program acts as a gateway or interface program between other, larger applications. CGI programs often are written in scripting languages such as Perl. Scripting languages really are not meant for large applications. You might create a program that translates and formats the data being sent to it from applications such as online catalogs, for example. This translated data then is passed to some type of database program. The database program does the necessary operations on its database and returns the results to your CGI program. Your CGI program then can reformat the returned data as needed for the Internet and return it to the online catalog customer, thus acting as a gateway between the HTML catalog, the HTTP request/response headers, and the database program.

How CGI Scripts Work : CGI can do so much because it is so simple. CGI is a very lightweight interface; it is essentially the minimum that the web server needs to provide in order to allow external processes to create web pages. Typically, when a web server gets a request for a static web page, the web server finds the corresponding HTML file on its filesystem. When a web server gets a request for a CGI script, the web server executes the CGI script as another process (i.e., a separate application); the server passes this process some parameters and collects its output, which it then returns to the client just as if had been fetched from a static file (see Figure).

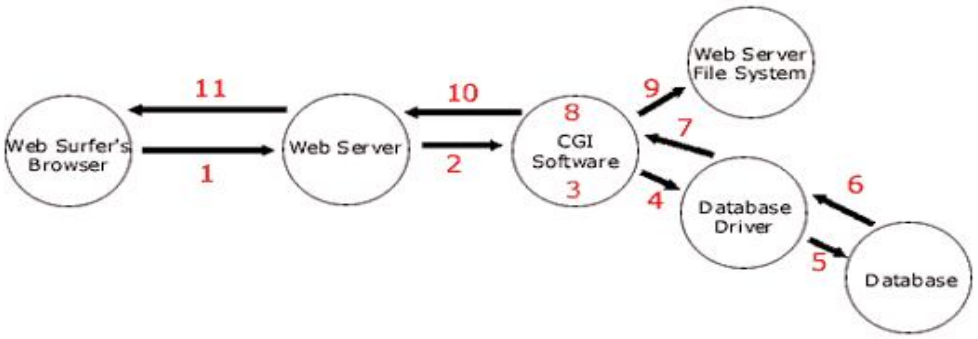


Fig. 7.1 Working of CGI Script.

1. The Web surfer fills out a form and clicks, "Submit." The information in the form is sent over the Internet to the Web server.

2. The Web server "grabs" the information from the form and passes it to the CGI software.
3. The CGI software performs whatever validation of this information that is required. For instance, it might check to see if an e-mail address is valid. If this is a database program, the CGI software prepares a database statement to add, edit, or delete information from the database.
4. The CGI software then executes the prepared database statement, which is passed to the database driver.
5. The database driver acts as a middleman and performs the requested action on the database itself.
6. The results of the database action are then passed back to the database driver.
7. The database driver sends the information from the database to the CGI software.
8. The CGI software takes the information from the database and manipulates it into the format that is desired.
9. If any static HTML pages need to be created, the CGI program accesses the Web server computer's file system and reads, writes, and/or edits files.
10. The CGI software then sends the result back to the Web Server.
11. The Web Server sends the results it got from CGI Software back to the Web Surfer's Browser.

7.2 Alternative Technologies

CGI programs written in Perl. Because Perl and CGI are so often used together, some people are unclear about the distinction. Perl is a programming language, and CGI is an interface that a program uses to handle requests from a web server. There are alternatives both to CGI and to Perl: there are new alternatives to CGI for handling dynamic requests, and CGI applications can be written in a variety of languages.

Why Perl? Although CGI applications can be written in any almost any language, Perl and CGI scripting have become synonymous to many programmers. As Hassan Schroeder, Sun's first webmaster, said in his oft-quoted statement, "Perl is the duct tape of the Internet." Perl is by far the most widely used language for CGI programming, and for many good reasons:

- Perl is easy to learn because it resembles other popular languages (such as C), because it is forgiving, and because when an error occurs it provides specific and detailed error messages to help you locate the problem quickly.
- Perl allows rapid development because it is interpreted; the source code does not need to be compiled before execution.
- Perl is easily portable and available on many platforms.
- Perl contains extremely powerful string manipulation operators, with regular expression matching and substitution built right into the language.

- Perl handles and manipulates binary data just as easily as it handles text.
- Perl does not require strict variable types; numbers, strings, and booleans are simply scalars.
- Perl interfaces with external applications very easily and provides its own filesystem functions.

Furthermore, Perl is fast. Perl isn't strictly an interpreted language. When Perl reads a source file, it actually compiles the source into low-level opcodes and then executes them. You do not generally see compilation and execution in Perl as separate steps because they typically occur together: Perl launches, reads a source file, compiles it, runs it, and exits. This process is repeated each time a Perl script is executed, including each time a CGI script is executed.

Alternatives to CGI : Several alternatives to CGI have appeared in recent years. They all build upon CGI's legacy and provide their own approaches to the same underlying goal: responding to queries and presenting dynamic content via HTTP. Most of them also attempt to avoid the main drawback to CGI scripts: creating a separate process to execute the script every time it is requested. Here is a list of some of the major alternatives to CGI:

ASP : Active Server Pages, or ASP, was created by Microsoft for its web server, but it is now available for many servers. The ASP engine is integrated into the web server so it does not require an additional process. It allows programmers to mix code within HTML pages instead of writing separate programs.

PHP : PHP is a programming language that is similar to Perl, and its interpreter is embedded within the web server. PHP supports embedded code within HTML pages. PHP is supported by the Apache web server.

ColdFusion : Allaire's ColdFusion creates more of a distinction than PHP between code pages and HTML pages. HTML pages can include additional tags that call ColdFusion functions. A number of standard functions are available with ColdFusion, and developers can create their own controls as extensions. The ColdFusion interpreter is integrated into the web server.

Java servlets : Java servlets were created by Sun. Servlets are similar to CGI scripts in that they are code that creates documents. However, servlets, because they use Java, must be compiled as classes before they are run, and servlets are dynamically loaded as classes by the web server when they are run. The interface is quite different than CGI. JavaServer Pages, or JSP, is another technology that allows developers to embed Java in web pages, much like ASP.

FastCGI : FastCGI maintains one or more instances of perl that it runs continuously along with an interface that allows dynamic requests to be passed from the web server to these instances. It avoids the biggest drawback to CGI, which is creating a new process for each request, while still remaining largely compatible with CGI. FastCGI is available for a variety of web servers.

mod_perl: *mod_perl* is a module for the Apache web server that also avoids creating separate instances of perl for each CGI. Instead of maintaining a separate instance of perl like FastCGI, *mod_perl* embeds the perl interpreter inside the web server.

This gives it a performance advantage and also gives Perl code written for *mod_perl* access to Apache's internals.

Despite a creation of these competing technologies, CGI continues to be the most popular method for delivering dynamic pages, and, despite what the marketing literature for some of its competitors may claim, CGI will not go away any time soon. Learning CGI teaches you how web transactions works at a basic level, which can only further your understanding of other technologies built upon this same foundation. Additionally, CGI is universal. Many alternative technologies require that you install a particular combination of technologies in addition to your web server in order to use them. CGI is supported by virtually every web server "right out of the box" and will continue to be that way far into the future.

7.3 URLs

In web terms, a resource represents anything available on the web, whether it be an HTML page, an image, a CGI script, etc. URLs provide a standard way to locate these resources on the Web. Note that URLs are not actually specific to HTTP; they can refer to resources in many protocols. Our discussion here will focus strictly on HTTP URLs.

What About URIs?

You may have also encountered the term URI and wondered about the difference between a URI and a URL. Actually, the terms are often interchangeable because all URLs are URIs. Uniform Resource Identifiers (URIs) are a more generalized class which includes URLs as well as Uniform Resource Names (URNs). A URN provides a name that sticks to an object even though the location of the object may move around. You can think of it this way: your name is similar to a URN, while your address is similar to a URL. Both serve to identify you in some way, and in this manner both are URIs.

Because URNs are just a concept and are not used on the Web today, you can safely think of URIs and URLs as interchangeable terms and not let the terminology throw you. Since we are not interested in other forms of URIs, we will try to avoid confusion altogether by just using the term URL in the text.

Elements of a URL : HTTP URLs consist of a scheme, a host name, a port number, a path, a query string, and a fragment identifier, any of which may be omitted under certain circumstances (see Figure: 2).

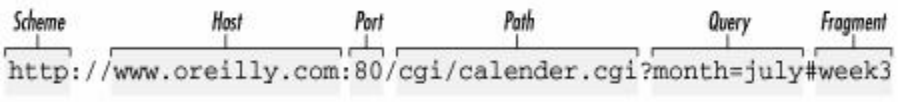


Fig. 7.2 :Components of a URL.

HTTP URLs contain the following elements:

Scheme : The scheme represents the protocol, and for our purposes will either be **http** or **https**. https represents a connection to a secure web server.

Host : The host identifies the machine running a web server. It can be a domain

name or an IP address, although it is a bad idea to use IP addresses in URLs and is strongly discouraged. The problem is that IP addresses often change for any number of reasons: a web site may move from one machine to another, or it may relocate to another network. Domain names can remain constant in these cases, allowing these changes to remain hidden from the user.

Port number : The port number is optional and may appear in URLs only if the host is also included. The host and port are separated by a colon. If the port is not specified, port 80 is used for http URLs and port 443 is used for https URLs.

Path information : Path information represents the location of the resource being requested, such as an HTML file or a CGI script. Depending on how your web server is configured, it may or may not map to some actual file path on your system. As we mentioned last chapter, the URL path for CGI scripts generally begin with /cgi/ or /cgi-bin/ and these paths are mapped to a similarly-named directory in the web server, such as */usr/local/apache/cgi-bin.*

Note that the URL for a script may include path information beyond the location of the script itself. For example, say you have a CGI at:

http://localhost/cgi/browse_docs.cgi

You can pass extra path information to the script by appending it to the end, for example:

http://localhost/cgi/browse_docs.cgi/docs/product/description.text

Here the path /docs/product/description.text is passed to the script. We explain how to access and use this additional path information in more detail in the next chapter.

Query string: A query string passes additional parameters to scripts. It is sometimes referred to as a search string or an index. It may contain name and value pairs, in which each pair is separated from the next pair by an ampersand (&), and the name and value are separated from each other by an equals sign (=).

Query strings can also include data that is not formatted as name-value pairs. If a query string does not contain an equal sign, it is often referred to as an index. Each argument should be separated from the next by an encoded space (encoded either as + or %20;) CGI scripts handle indexes a little differently.

Fragment identifier: Fragment identifiers refer to a specific section in a resource. Fragment identifiers are not sent to web servers, so you cannot access this component of the URLs in your CGI scripts. Instead, the browser fetches a resource and then applies the fragment identifier to locate the appropriate section in the resource. For HTML documents, fragment identifiers refer to anchor tags within the document:

```
<a name="anchor" >Here is the content you're after...</a>
```

The following URL would request the full document and then scroll to the section marked by the anchor tag:

http://localhost/document.html#anchor

Web browsers generally jump to the bottom of the document if no anchor for the fragment identifier is found.

Absolute and Relative URLs : Many of the elements within a URL are optional. You

may omit the scheme, host, and port number in a URL if the URL is used in a context where these elements can be assumed. For example, if you include a URL in a link on an HTML page and leave out these elements, the browser will assume the link applies to a resource on the same machine as the link. There are two classes of URLs:

Absolute URL : URLs that include the hostname are called absolute URLs. An example of an absolute URL is

http://localhost/cgi/script.cgi.

Relative URL : URLs without a scheme, host, or port are called relative URLs. These can be further broken down into full and relative paths:

Full paths : Relative URLs with an absolute path are sometimes referred to as full paths (even though they can also include a query string and fragment identifier). Full paths can be distinguished from URLs with relative paths because they always start with a forward slash. Note that in all these cases, the paths are virtual paths, and do not necessarily correspond to a path on the web server's filesystem. An example of an absolute path is ***/index.html***.

Relative Paths : Relative URLs that begin with a character other than a forward slash are relative paths. Examples of relative paths include

`script.cgi and ../images/photo.jpg.`

7.4 THE HYPERTEXT TRANSPORT PROTOCOL

The Hypertext Transfer Protocol (HTTP) is an application-level protocol with the lightness and speed necessary for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol which can be used for many tasks, such as name servers and distributed object management systems, through extension of its request methods (commands). A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred.

In the general form HTTP is the protocol that clients and servers use to communicate on the Web. HTTP is the underlying mechanism on which CGI operates, and it directly determines what you can and cannot send or receive via CGI.

7.4.1 HTTP Properties

1. A comprehensive addressing scheme: The HTTP protocol uses the concept of reference provided by the Universal Resource Identifier (URI) as a location (URL) or name (URN), for indicating the resource on which a method is to be applied. When an HTML hyperlink is composed, the URL (Uniform Resource Locator) is of the general form http://host:port-number/path/file.html.

2. Client-Server Architecture: The HTTP protocol is based on a request/response paradigm. The communication generally takes place over a TCP/IP connection on the Internet. The default port is 80, but other ports can be used.

3. The HTTP protocol is connectionless and stateless: After the server has

responded to the client's request, the connection between client and server is dropped and forgotten. There is no "memory" between client connections. The pure HTTP server implementation treats every request as if it was brand-new, i.e. without context.

4. An extensible and open representation for data types: HTTP uses Internet Media Types (formerly referred to as MIME Content-Types) to provide open and extensible data typing and type negotiation. When the HTTP Server transmits information back to the client, it includes a MIME-like (Multipart Internet Mail Extension) header to inform the client what kind of data follows the header.

7.4.2 The Request and Response Cycle of HTTP

When a web browser requests a web page, it sends a request message to a web server. The message always includes a header, and sometimes it also includes a body. The web server in turn replies with a reply message. This message also always includes a header and it usually contains a body.

There are two features that are important in understanding HTTP:

- It is a request/response protocol: each response is preceded by a request.
- Although requests and responses each contain different information, the header/body structure is the same for both messages. The header contains meta-information -- information about the message -- and the body contains the content of the message.

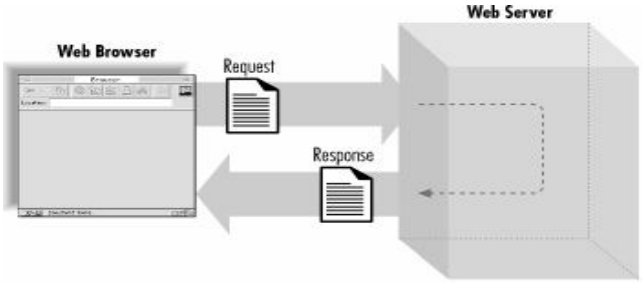


Fig.7.3 : The HTTP request / response cycle.

Figure shows an example of an HTTP transaction. Say you told your browser you wanted a document at *http:///localhost/index.html*. The browser would connect to the machine at localhost on port 80 and send it the following message:

7.5 BROWSER REQUESTS

Every HTTP interaction starts with a request from a client, typically a web browser. A user provides a URL to the browser by typing it in, clicking on a hyperlink, or selecting a bookmark, and the browser fetches the corresponding document. To do that, it must create an HTTP request (see Figure 5).

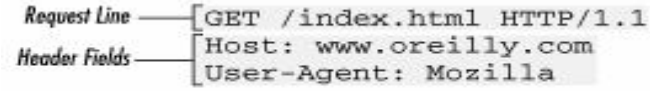


Fig. 7.4 : The structure of HTTP request headers.

Recall that in our previous example, a web browser generated the following request when it was asked to fetch the URL http://localhost/index.html :

```
GET /index.html HTTP/1.1
Host: localhost
Accept: image/gif, image/x-xbitmap, image/jpeg */*
Accept-Language: en
Connection: Keep-Alive
User-Agent: Mozilla/4.0 (compatible; MSIE 4.5; Mac_PowerPC)
```

Fig. 7.5 : The request line.

From our discussion of URLs, you know that the URL can be broken down into multiple elements. The browser creates a network connection by using the hostname and the port number (80 by default). The scheme (http) tells our web browser that it is using the HTTP protocol, so once the connection is established, it sends an HTTP request for the resource. The first line of an HTTP request is the request line, which includes a full virtual path and query string (if present); see Figure 6.

7.5.1 The Request Line

The first line of an HTTP request includes the request method, a URL to the resource being requested, and the version string of the protocol. Request methods are case-sensitive and uppercase. There are several request methods defined by HTTP although a web server may not make all of them available for each resource (see Table 1). The version string is the name and version of the protocol separated by a slash. HTTP 1.0 and HTTP 1.1 are represented as HTTP/1.0 and HTTP/1.1. Note that https requests also produce one of these two HTTP protocol strings.

Method	Description
GET	Asks the server for the given resource
HEAD	Used in the same cases that a GET is used but it only returns HTTP headers and no content
POST	Asks the server to modify information stored on the server
PUT	Asks the server to create or replace a resource on the server
DELETE	Asks the server to delete a resource on the server
CONNECT	Used to allow secure SSL connections to tunnel through HTTP connections
OPTIONS	Asks the server to list the request methods available for the given resource
TRACE	Asks the server to echo back the request headers as it received them

Table 7.1 : HTTP Request Methods.

7.5.2 Request Header Field Lines

The client generally sends several header fields with its request. As mentioned earlier, these consist of a field name, a colon, some combination of spaces or tabs

(although one space is most common), and a value (see Figure 7). These fields are used to pass additional information abouthe request or about the client, or to add conditions to the request. We'll discuss the common browser headers here; they are listed in Table 2. Those connected with content negotiation and caching are discussed later in this chapter.

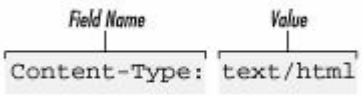


Fig. 7.6 : A header field line.

Header	Description
Host	Specifies the target hostname
Content-Length	Specifies the length (in bytes) of the request content
Content-Type	Specifies the media type of the request
Authentication	Specifies the username and password of the user requesting the resource
User-Agent	Specifies the name, version, and platform of the client
Referer	Specifies the URL that referred the user to the current resource
Cookie	Returns a name/value pair set by the server on a previous response

Table 7.2 : Common HTTP Request Headers.

7.6 SERVER RESPONSES

Server responses, like client requests, always contain HTTP headers and an optional body. Here is the server response from our earlier example:

```
HTTP/1.1 200 OK
Date: Sat, 18 Mar 2000 20:35:35 GMT
Server: Apache/1.3.9 (Unix)
Last-Modified: Wed, 20 May 1998 14:59:42 GMT
ETag: "74916-656-3562efde”
Content-Length: 141
Content-Type: text/html
<HTML>
<HEAD><TITLE>Sample Document</TITLE></HEAD>
<BODY>
<H1>Sample Document</H1>
<P>This is a sample HTML document!</P> </BODY> </HTML>
```

The structure of the headers for the response is the same as for requests. The first header line has a special meaning, and is referred to as the status line. The remaining lines are name-value header field lines. See Figure 8.

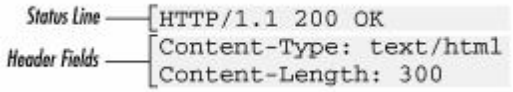


Fig. 7.7 : The structure of an HTTP response header.

7.6.1 The Status Line

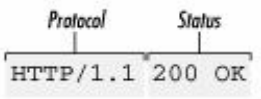


Fig. 7.8

The first line of the header is the status line, which includes the protocol and version just as in HTTP requests, except that this information comes at the beginning instead of at the end. This string is followed by a space and the three-digit status code, as well as a text version of the status. See Figure.

Web servers can send any of dozens of status codes. For example, the server returns a status of 404 Not Found if a document doesn't exist and 301 Moved Permanently if a document is moved. Status codes are grouped into five different classes according to their first digit:

1. **1xx :** These status codes were introduced for HTTP 1.1 and used at a low level during HTTP transactions. You won't use 100-series status codes in CGI scripts.
2. **2xx :** 200-series status codes indicate that all is well with the request.
3. **3xx :** 300-series status codes generally indicate some form of redirection. The request was valid, but the browser should find the content of its response elsewhere.
4. **4xx :** 400-series status codes indicate that there was an error and the server is blaming the browser for doing something wrong.
5. **5xx :** 500-series status codes also indicate there was an error, but in this case the server is admitting that it or a CGI script running on the server is the culprit.

7.6.2 Server Headers

After the status line, the server sends its HTTP headers. Some of these server headers are the same headers that browsers send with their requests. The common server headers are listed in table 3.

Header	Description
Content-Base	Specifies the base URL for resolving all relative URLs within the document
Content-Length	Specifies the length (in bytes) of the body
Content-Type	Specifies the media type of the body
Date	Specifies the date and time when the response was sent
ETag	Specifies an entity tag for the requested resource
Last-Modified	Specifies the date and time when the requested resource was last modified
Location	Specifies the new location for the resource
Server	Specifies the name and version of the web server
Set-Cookie	Specifies a name-value pair that the browser should provide with future requests
WWW-Authenticate	Specifies the authorization scheme and realm

Table 7.3 : Common HTTP Server Headers

7.7 PROXIES

Web browsers do not interact directly with web servers; instead they communicate via a proxy. HTTP proxies are often used to reduce network traffic,

allow access through firewalls, provide content filtering, etc. Proxies have their own functionality that is defined by the HTTP standard. You can think of a proxy as a combination of a simplified client and a server (see Figure 10). An HTTP client connects to a proxy with a request; in this way, it acts like a server. The proxy forwards the request to a web server and retrieves the appropriate response; in this way, it acts like a client. Finally, it fulfills its server role by returning the response to the client. Figure shows how an HTTP proxy affects the request and response cycle. Note that although there is only one proxy represented here, it's quite possible for a single HTTP transaction to pass through many proxies.

Proxies : Proxies affect us in two ways. First, they make it impossible for web servers to reliably identify the browser. Second, proxies often cache content. When a client makes a request, proxies may return a previously cached response without contacting the target web server.

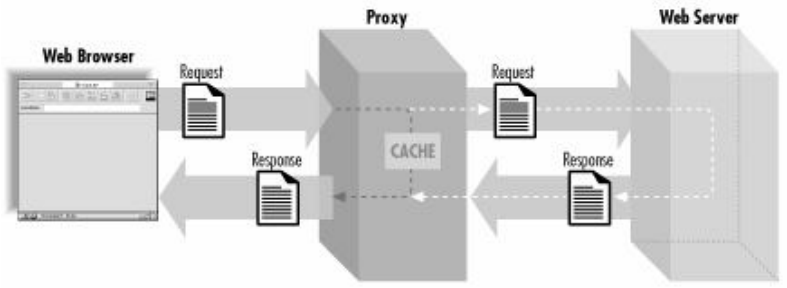


Fig. 7.9 : HTTP proxies and the request / response cycle.

7.7.1 Identifying Clients

Basic HTTP requests do not contain any information that identifies the client. In a simple network transaction, this is generally not an issue, because a server knows which client is talking to it. We can see this by analogy. If someone walks up to you and hands you a note, you know who delivered the note regardless of what the note says. It's apparent from the context.

The problem is determining who wrote the note. If the note isn't signed, you may not know whether the person handing you the note wrote the note or is simply delivering it. The same is true in HTTP transactions. Web servers know which system is requesting information from them, but they don't know whether this client is a web browser that originated the request (i.e., the author of the note) or if they are just a proxy (i.e., the messenger). This is not a shortcoming of proxies, because this anonymity is actually a feature of proxies integrated into firewalls. Organizations with firewalls typically prefer that the outside world not know the addresses of systems behind their firewall.

7.7.2 Caching

One of the benefits of proxies is that they make HTTP transactions more efficient by sharing some of the work the web server typically does. Proxies accomplish this by caching requests and responses. When a proxy receives a request, it checks its

cache for a similar, previous request. If it finds this, and if the response is not stale (out of date), then it returns this cached response to the client. The proxy determines whether a response is stale by looking at HTTP headers of the cached response, by sending a HEAD request to the target web server to retrieve new headers to compare against, and via its own algorithms. Regardless of how it determines this, if the proxy does not need to fetch a new, full response from the target web server, the proxy reduces the load on the server and reduces network traffic between the server and itself. This can also make the transaction much faster for the user.

Because most resources on the Internet are static HTML pages and images that do not often change, caching is very helpful. For dynamic content, however, caching can cause problems. CGI scripts allow us to generate dynamic content; a request to one CGI script can generate a variety of responses. Imagine a simple CGI script that returns the current time. The request for this CGI script looks the same each time it is called, but the response should be different each time. If a proxy caches the response from this CGI script and returns it for future requests, the user would get an old copy of the page with the wrong time. Fortunately, there are ways to indicate that the response from a web server should not be cached. We'll explore this in the next chapter. HTTP 1.1 also added specific guidelines for proxies that solved a number of problems with earlier proxies. Most current proxies, even if they do not fully implement HTTP 1.1, have adopted these guidelines.

Caching is not unique to proxies. You probably know that browsers do their own caching too. Some web pages have instructions telling users to clear their web browser's cache if they are having problems receiving up-to-date information. Proxies present a challenge because users cannot clear the cache of intermediate proxies (they often may not even know they are using a proxy) as they can for their browser.

7.8 CONTENT NEGOTIATION

Content negotiation is a mechanism defined in the HTTP specification that makes it possible to serve different versions of a document (or more generally, a resource) at the same URI, so that user agents can specify which version fit their capabilities the best. One of the most classical uses of this mechanism is to serve an image in GIF or PNG format, so that a browser that doesn't understand PNG (e.g. MS Internet Explorer) can still displays the GIF version. To summarize how this works, when a user agent submits a request to a server, the user agent informs the server what media types the user agent understands along with indications of how well it understands them. More precisely, the user agent uses an Accept HTTP header that lists acceptable media types. The server is then able to supply the version of the resource that best fits the user agent's needs.

So, a resource may be available in several different representations. For example, it might be available in different languages or different media types, or a combination. One way of selecting the most appropriate choice is to give the user an index page, and let them select. However it is often possible for the server to choose automatically. This works because browsers can send as part of each request information about the representations they prefer. For example, a browser could indicate that it would like to see information in French, if possible, else English will do. Browsers indicate their preferences by headers in the request. To request only French representations, the browser would send.

SHORT QUESTION ANSWERS

Q.1 What is CGI?

Ans. CGI, or Common Gateway Interface, is the standard programming interface between web servers and external programs. The CGI standard lets web browsers pass information to programs written in any language. The CGI standard does not exist in isolation; it is ependent on the HTML and HTTP standards. HTML is the standard that lets web browsers understands document content. HTTP is the communications protocol that, among other things, lets web servers talk with web browser.

CGI gives you a way to make web sites dynamic and interactive.

- Common Gateway Interface" means:-
- Common - interacts with many different operating systems.
- Gateway - provides users with a way to gain access to different programs, like databases or picture generators.
- Interface - uses a well-defined method to interact with a web server.

CGI applications should be designed to take advantage of the centralized nature of a web server. They are great for searching databases, processing HTML form data, and other applications that require limited interaction with a user.

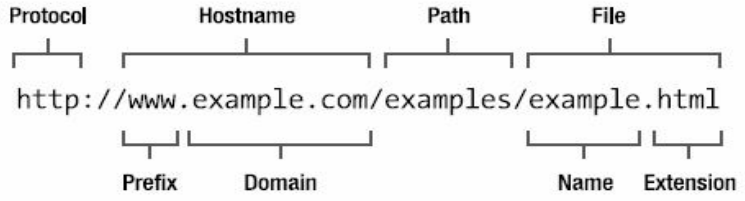
Q.2 How does Common Gateway work?

Ans. Following are the steps that shows how CGI scripts works:

1. The Web surfer fills out a form and clicks, "Submit." The information in the form is sent over the Internet to the Web server.
2. The Web server "grabs" the information from the form and passes it to the CGI software.
3. The CGI software performs whatever validation of this information that is required. For instance, it might check to see if an e-mail address is valid. If this is a database program, the CGI software prepares a database statement to add, edit, or delete information from the database.
4. The CGI software then executes the prepared database statement, which is passed to the database driver.
5. The database driver acts as a middleman and performs the requested action on the database itself.
6. The results of the database action are then passed back to the database driver.
7. The database driver sends the information from the database to the CGI software.
8. The CGI software takes the information from the database and manipulates it into the format that is desired.
9. If any static HTML pages need to be created, the CGI program accesses the Web server computer's file system and reads, writes, and/or edits files.
10. The CGI software then sends the result back to the Web Server.
11. The Web Server sends the results it got from CGI Software back to the Web Surfer's Browser.

Q.3 What is URL ? What are different components of URL ?

Ans. Uniform Resource Locator (URL) is the global address of documents and other resources on the World Wide Web. Uniform Resource Locator is the address that defines the route to a file on an Internet server (Web server, FTP server, mail server, etc.). URLs are typed into a Web browser to access Web pages and files, and URLs are embedded within the pages themselves as hypertext links.



Q.4 What is HTTP ?

Ans. The Hypertext Transfer Protocol (HTTP) is an application-level protocol with the lightness and speed necessary for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol which can be used for many tasks, such as name servers and distributed object management systems, through extension of its request methods (commands). A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred.

In the general form HTTP is the protocol that clients and servers use to communicate on the Web. HTTP is the underlying mechanism on which CGI operates, and it directly determines what you can and cannot send or receive via CGI.

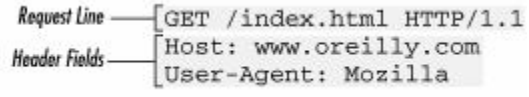
Q.5 What do you mean by Server-side?

Ans. Occurring on the server side of a client-server system. For example, on the World Wide Web, CGI scripts are server-side applications because they run on the Web server. In contrast, JavaScript scripts are client-side because they are executed by your browser (the client). Java applets can be either server-side or client- side depending on which computer (the server or the client) executes them.

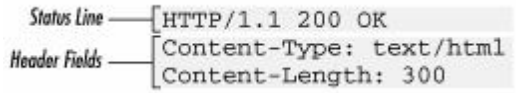
Q.6 What is Request and Response cycle of HTTP ?

Ans. When a web browser requests a web page, it sends a request message to a web server. The message always includes a header, and sometimes it also includes a body. The web server in turn replies with a reply message. This message also always includes a header and it usually contains a body.

Browser Requests : Every HTTP interaction starts with a request from a client, typically a web browser. A user provides a URL to the browser by typing it in, clicking on a hyperlink, or selecting a bookmark, and the browser fetches the corresponding document. To do that, it must create an HTTP request (see Fig.5).

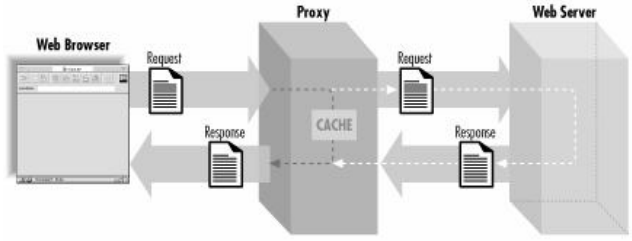


Server Responses: Server responses, like client requests, always contain HTTP headers and an optional body. The structure of the headers for the response is the same as for requests. The first header line has a special meaning, and is referred to as the status line. The remaining lines are name-value header field lines. See Figure.



Q.7 What is Proxy Server ?

Ans. Web browsers do not interact directly with web servers; instead they communicate via a proxy. HTTP proxies are often used to reduce network traffic, allow access through firewalls, provide content filtering, etc. Proxies have their own functionality that is defined by the HTTP standard. You can think of a proxy as a combination of a simplified client and a server (see Figure 10). An HTTP client connects to a proxy with a request; in this way, it acts like a server. The proxy forwards the request to a web server and retrieves the appropriate response; in this way, it acts like a client. Finally, it fulfills its server role by returning the response to the client. Figure shows how an HTTP proxy affects the request and response cycle. Note that although there is only one proxy represented here, it's quite possible for a single HTTP transaction to pass through many proxies.



Q.8 What is meant by Stateless Connection?

Ans. When a web server receives a HTTP request from a web browser it evaluates the request and returns the requested document, if it exists, and then breaks the HTTP connection. This document is preceded by the response header, which has details about how to display the document that will be sent by the server. Each time a request is made to the server, it is as if there was no prior connection and each request can yield only a single document. This is known as Stateless Connection.

MULTIPLE CHOICE QUESTIONS

1. Full form of CGI

a) Computer Graphics Interface b) Common Gateway Interface

c) Common Graphics Interface d) Computer Gateway Interface

2. The different ways to send data to the CGI program

a) Arguments of the CGI program b) Environment variables

c) Standard input d) All of these

3. CGI is a set of rules that works as an _____

a) Interface between web server b) External programs on the web server

c) Both a & b d) None of these

4. _____ works as an intermediary between the user and the server where direct access of information from the server in not readable by the user

a) Forms b) Status

c) Location d) Gateways

5. _____ are created dynamically in response to user request:

a) Virtual Documents b) Dynamic document

c) Both of these d) None if these

6. It contains a number of graphical widgets to get the information for the user:

a) Radio button b) Text fields

c) Check boxes d) All of these

7. The major application of CGI are forms

a) Gateways b) Dynamic document

c) Both of these d) None of these

KEY TO OBJECTIVE QUESTIONS

1.b 2.d 3.c 4.d 5.a 6.d 7.c

IMPORTANT QUESTIONS

Q.1 What is gateway interface ? Explain.

Q.2 What is the need of Sever Side Scripting on the web ?

Q.3 Compare CGI, ASP and JSP server side technology.

Q.4 How CGI works ? What is CGI Environment (Explain with components) ?

Q.5 What is HTTP ? How it works ?

Q.6 What is URL ? What are different fields ? Explain them.

Q.7 Explain the process of Browser request & Server response.

Q.8 How we can use HTML tags for writing forms.

Q.9 Explain the following :

(a) URL

(b) HTTP

(c) Proxy server

****

CGI ENVIRONMENT USING PERL

IN THIS CHAPTER, YOU WILL LEARN

- The CGI Environment
- CGI Environment Variables
- CGI Output
- Forms and CGI
- Sending Data to Server
- Decoding Form Input
- Efficiency and Optimization
- Guidelines for Better CGI Applications

Common Gateway Interface (CGI) is a standard for interfacing external programs with information servers on the Internet. So what does this mean? Basically, CGI is distinguished from a plain HTML document in that the plain HTML document is static, while CGI executes in real-time to output dynamic information. A program that implements CGI is executable, while the plain HTML document exists as a constant text file that doesn't change. CGI, then, obtains information from users and tailors pages to their needs. While there are newer ways to perform the same kinds of actions that traditionally have been implemented with CGI, the latter is older and, in many ways, more versatile. It is for this reason that, over time, CGI has become generalized to refer to any program that runs on a Web server and interacts with a browser.

For example, if you wanted to allow people from all over the world to query some database you had developed, you could create an executable CGI script that would transmit information to the database engine and then receive results and display them in the user's Web browser. The user could not directly access the database without some gateway to allow access. This link between the database and the user is the "gateway," which is where the CGI standard originated.

A CGI script can be written in any language that allows it to be executed (e.g., C/C++, Fortran, PERL, TCL, Visual Basic, AppleScript, Python, and Unix shells), but by far, the most common language for CGI scripting is PERL, followed by C/C++. A CGI script is easier to debug, modify, and maintain than the typical compiled program, so many people prefer CGI for this reason.

8.1 The CGI Environment

CGI establishes a particular environment in which CGI scripts operate. This environment includes such things as what current working directory the script starts

in, what variables are preset for it, where the standard file handles are directed, and so on. In return, CGI requires that scripts be responsible for defining the content of the HTTP response and at least a minimal set of HTTP headers.

8.1.1 File Handles

Perl scripts generally start with three standard file handles predefined: STDIN, STDOUT, and STDERR. CGI Perl scripts are no different. These file handles have particular meaning within a CGI script, however.

1. **STDIN :** When a web server receives an HTTP request directed to a CGI script, it reads the HTTP headers and passes the content body of the message to the CGI script on STDIN. Because the headers have already been removed, STDIN will be empty for GET requests that have no body and contain the encoded form data for POST requests. Note that there is no end-of-file marker, so if you try to read more data than is available, your CGI script will hang, waiting for more data on STDIN that will never come (eventually, the web server or browser should time out and kill this CGI script but this wastes system resources). Thus, you should never try to read from STDIN for GET requests. For POST requests, you should always refer to the value of the Content-Length header and read only that many bytes.
2. **STDOUT :** Perl CGI scripts return their output to the web server by printing to STDOUT. This may include some HTTP headers as well as the content of the response, if present. Perl generally buffers output on STDOUT and sends it to the web server in chunks. The web server itself may wait until the entire output of the script has finished before sending it onto the client. For example, the iPlanet (formerly Netscape) Enterprise Server buffers output, while Apache (1.3 and higher) does not.
3. **STDERR :** CGI does not designate how web servers should handle output to STDERR, and servers implement this in different ways, but they almost always produces a 500 Internal Server Error reply. Some web servers, like Apache, append STDERR output to the web server's error log, which includes other errors such as authorization failures and requests for documents not on the server. This is very helpful for debugging errors in CGI scripts.

8.1.2 Environment Variables

CGI scripts are given predefined environment variables that provide information about the web server as well as the client. Much of this information is drawn from the headers of the HTTP request. In Perl, environment variables are available to your script via the global hash %ENV. You are free to add, delete, or change any of the values of %ENV. Subprocesses created by your script will also inherit these environment variables, along with any changes you've made to them.

8.2 CGI ENVIRONMENT VARIABLES

The standard CGI environment variables listed below should be available on any server supporting CGI. Nonetheless, if you loop through all the keys in %ENV, you will probably not see all the variables listed here. If you recall, some HTTP request headers are used only with certain requests. For example, the Content-length header is sent only with POST requests. The following list contains the variables documented in the CGI/1.1 specification and some variables commonly set by web servers. These environment variables are set when the server executes the gateway program. The following environment variables are not request-specific and are set for all requests:

1. **SERVER_SOFTWARE:** The name and version of the information server software

answering the request (and running the gateway). Format: name/version.

2. **SERVER_NAME:** The server's hostname, DNS alias, or IP address as it would appear in self-referencing URLs.
3. **GATEWAY_INTERFACE:** The revision of the CGI specification to which this server complies. Format: CGI/revision

   The following environment variables are specific to the request being fulfilled by the gateway program:

4. **SERVER_PROTOCOL:** The name and revision of the information protcol this request came in with. Format: protocol/revision
5. **SERVER_PORT:** The port number to which the request was sent.
6. **REQUEST_METHOD:** The method with which the request was made. For HTTP, this is "GET", "HEAD", "POST", etc.
7. **PATH_INFO:** The extra path information, as given by the client. In other words, scripts can be accessed by their virtual pathname, followed by extra information at the end of this path. The extra information is sent as PATH_INFO. This information should be decoded by the server if it comes from a URL before it is passed to the CGI script.
8. **PATH_TRANSLATED:** The server provides a translated version of PATH_INFO, which takes the path and does any virtual-to-physical mapping to it.\
9. **SCRIPT_NAME:** A virtual path to the script being executed, used for self-referencing URLs.
10. **QUERY_STRING:** The information which follows the ? in the URL which referenced this script. This is the query information. It should not be decoded in any fashion. This variable should always be set when there is query information, regardless of command line decoding.
11. **REMOTE_HOST :** The hostname making the request. If the server does not have this information, it should set REMOTE_ADDR and leave this unset.
12. **REMOTE_ADDR :** The IP address of the remote host making the request.
13. **AUTH_TYPE :** If the server supports user authentication, and the script is protects, this is the protocol-specific authentication method used to validate the user.
14. **REMOTE_USER :** If the server supports user authentication, and the script is protected, this is the username they have authenticated as.

15 **REMOTE_IDENT :** If the HTTP server supports RFC 931 identification, then this variable will be set to the remote user name retrieved from the server. Usage of this variable should be limited to logging only.

15. **CONTENT_TYPE :** For queries which have attached information, such as HTTP POST and PUT, this is the content type of the data.
16. **CONTENT_LENGTH :** The length of the said content as given by the client.

8.3 CGI OUTPUT

Script output : The script sends its output to stdout. This output can either be a document generated by the script, or instructions to the server for retrieving the desired output.

Script Naming Conventions : Normally, scripts produce output which is interpreted and sent back to the client. An advantage of this is that the scripts do not need to send a full HTTP/1.0 header for every request.

Parsed headers : The output of scripts begins with a small header. This header consists of text lines, in the same format as an HTTP header, terminated by a blank line (a line with only a linefeed or CR/LF).

Any headers which are not server directives are sent directly back to the client. Currently, this specification defines three server directives:

Content-type : This is the MIME type of the document you are returning.

Location : This is used to specify to the server that you are returning a reference to a document rather than an actual document.

- If the argument to this is a URL, the server will issue a redirect to the client.
- If the argument to this is a virtual path, the server will retrieve the document specified as if the client had requested that document originally. ? directives will work in here, but # directives must be redirected back to the client.

Status : This is used to give the server an HTTP/1.0 status line to send to the client. The format is nnn xxxxx, where nnn is the 3-digit status code, and xxxxx is the reason string, such as "Forbidden".

Examples : Let's say I have a fromgratz to HTML converter. When my converter is finished with its work, it will output the following on stdout (note that the lines beginning and ending with --- are just for illustration and would not be output):

--- start of output ---

Content-type: text/html

--- end of output ---

8.4 FORMS AND CGI

HTML forms are the user interface that provides input to your CGI scripts. They are primarily used for two purposes: collecting data and accepting commands. Examples of data you collect may include registration information, payment information, and online surveys. You may also collect commands via forms, such as using menus, checkboxes, lists, and buttons to control various aspects of your application. In many cases, your forms will include elements for both: collecting data as well as application control.

A great advantage of HTML forms is that you can use them to create a frontend for numerous gateways (such as databases or other information servers) that can be accessed by any client without worrying about platform dependency.

In order to process data from an HTML form, the browser must send the data via an HTTP request. A CGI script cannot check user input on the client side; the user must press the submit button and the input can only be validated once it has travelled to the server. JavaScript, on the other hand, can perform actions in the browser. It can be used in conjunction with CGI scripts to provide a more responsive user interface.

Form Tags : A form consists of two distinct parts – The HTML code and the CGI

program. HTML tags create the visual representation of the form, while the CGI program decodes (or processes) the information contained within the form. Before we look at how CGI programs process form information, let's understand how a form is created. In this section, we'll cover the form tags and show examples of their use.

<FORM ACTION="/cgi-bin/program.pl" METHOD="POST">

The <FORM> tag starts the form. A document can consist of multiple forms, but forms cannot be nested; a form cannot be placed inside another form.

8.5 SENDING DATA TO SERVER

There are two methods for sending form data: GET and POST. The main difference between these methods is the way in which the form data is passed to the CGI program. If the GET method is used, the query string is simply appended to the URL of the program when the client issues the request to the server. This query string can then be accessed by using the environment variable QUERY_STRING. Here is a sample GET request by the client, which corresponds to the first form example:

```
GET /cgi-bin/program.pl?user=Larry%20Bird&age=35&pass=testing HTTP/1.0
Accept: www/source
Accept: text/html
Accept: text/plain
User-Agent: Lynx/2.4 libwww/2.14
```

The query string is appended to the URL after the "?" character. The server then takes this string and assigns it to the environment variable QUERY_STRING. The information in the password field is not encrypted in any way; it is plain text. You have to be very careful when asking for sensitive data using the password field. If you want security, please use server authentication. The GET method has both advantages and disadvantages. The main advantage is that you can access the CGI program with a query without using a form. In other words, you can create " canned queries." Basically, you are passing parameters to the program. For example, if you want to send the previous query to the program directly, you can do this:

```
<A HREF="/cgi-bin/program.pl?user=Larry%20Bird&age=35&pass=testing">CGI Program</A>
```

Here is a simple program that will aid you in encoding data:

```
#!/usr/local/bin/perl
print "Please enter a string to encode: ";
$string = </p>
<STDIN>;
chop ($string);
$string =~ s/(\W)/sprintf("%%%x", ord($1))/eg;
print "The encoded string is: ", "\n";
print $string, "\n";
exit(0);
```

This is not a CGI program; it is meant to be run from the shell. When you run the program, the program will prompt you for a string to encode. The <STDIN> operator reads one line from standard input. It is similar to the <FILEHANDLE> construct we have been using. The chop command removes the trailing newline character ("\n") from the input string. Finally, the user-specified string is converted to a hexadecimal value with the sprintf command, and printed out to standard output. A query is one method of passing information to a CGI program via the URL. The other method involves sending extra path information to the program. Here is an example:

```
<A HREF="/cgi-bin/program.pl/user=Larry%20Bird/age=35/pass=testing>CGI Program</A>
```

The string "/user=Larry%20Bird/age=35/pass=testing" will be placed in the environment variable PATH_INFO when the request gets to the CGI program. This method of passing information to the CGI program is generally used to provide file information, rather than form data. The NCSA imagemap program works in this manner by passing the filename of the selected image as extra path information.

If you use the "question-mark" method or the pathname method to pass data to the program, you have to be careful, as the browser or the server may truncate data that exceeds an arbitrary number of characters.

Now, here is a sample POST request:

```
POST /cgi-bin/program.pl HTTP/1.0Accept: www/source
Accept: text/html
Accept: text/plain
User-Agent: Lynx/2.4 libwww/2.14
Content-type: application/x-www-form-urlencoded
Content-length: 35
user=Larry%20Bird&age=35&pass=testing
```

The main advantage to the POST method is that query length can be unlimited--you don't have to worry about the client or server truncating data. To get data sent by the POST method, the CGI program reads from standard input. However, you cannot create "canned queries."

8.6 DECODING FORM INPUT

In order to access the information contained within the form, a decoding protocol must be applied to the data. First, the program must determine how the data was passed by the client. This can be done by examining the value in the environment variable REQUEST_METHOD. If the value indicates a GET request, either the query string or the extra path information must be obtained from the environment variables. On the other hand, if it is a POST request, the number of bytes specified by the CONTENT_LENGTH environment variable must be read from standard input. The algorithm for decoding form data follows:

1) Determine request protocol (either GET or POST) by checking the REQUEST_METHOD environment variable.
2) If the protocol is GET, read the query string from QUERY_STRING and/or the

extra path information from PATH_INFO.

3) If the protocol is POST, determine the size of the request using CONTENT_LENGTH and read that amount of data from the standard input.

4) Split the query string on the "&" character, which separates key-value pairs (the format is key=value&key=value...).

5) Decode the hexadecimal and "+" characters in each key-value pair.

6) Create a key-value table with the key as the index.

You might wonder why a program needs to check the request protocol, when you know exactly what type of request the form is sending. The reason is that by designing the program in this manner, you can use one module that takes care of both types of requests. It can also be beneficial in another way.

Say you have a form that sends a POST request, and a program that decodes both GET and POST requests. Suppose you know that there are three fields: user, age, and pass. You can fill out the form, and the client will send the information as a POST request. However, you can also send the information as a query string because the program can handle both types of requests; this means that you can save the step of filling out the form. You can even save the complete request as a hotlist item, or as a link on another page.

8.7 EFFICIENCY AND OPTIMIZATION

CGI applications, run under normal conditions, are not exactly speed demons. Let's try to understand why CGI applications are so slow. When a user requests a resource from a web server that turns out to be a CGI application, the server has to create another process to handle the request. And when you're dealing with applications that use interpreted languages, like Perl, there is an additional delay incurred in firing up the interpreter, then parsing and compiling the application.

So, how can we possibly improve the performance of Perl CGI applications? We could ask Perl to interpret only the most commonly used parts of our application and delay interpreting other pieces unless necessary. That certainly would speed up applications. Or, we could turn our application into a server (daemon) that runs in the background and executes on demand.

Perl Tips : Using these, you can improve the performance of your CGI applications.

1).Benchmark your code
2).Benchmark modules
3).Localize variables with my
4).Avoid slurping data from files
5).Clear arrays with undef instead of ( )
6).Use SelfLoader where applicable
7).Use autouse where applicable
8).Avoid the shell
9).Find existing solutions for your problems
10).Optimize your regular expressions

8.8 Guidelines for Better CGI Applications

CGI developers set guidelines that help their code. In a corporate setting, these

guidelines tend to become the standards through which teams of developers understand how to easily read the code that their neighbors produce. There are two types of guidelines:

- Architectural Guidelines
- Coding Guidelines

8.8.1 Architectural Guidelines

Architectural Guidelines include the following tips on how to architect a CGI application.

- **Plan for Future Growth :** Web sites may start small, but they typically grow and evolve over time. You may start out working on a small site without many developers where it is easy to coordinate work. However, as web sites grow and the staff that develops and supports the web site grows, it becomes more critical that it is designed well. Developers should have a development site where they can work on their own copies of the web site without affecting the production web server.

- **Use Directories to Organize Your Projects :** You should develop a directory structure that helps you organize information easily. Foer examplae, For example, if you had an web storefront application, you might store the components in subdirectories within ***/usr/local/projects/web_store*** like so:

- **Use Relative URLs :** Your web site will be most flexible if you use relative URLs instead of absolute URLs. In other words, do not include the domain name of your web server when you do not need to. If your development and production web servers have different names, you want your code to work on either system with very little reconfiguration.

- **Separate Configuration from Your Primary Code :** Information that is likely to change in the program or that is dependent upon the environment should be placed in a separate setup file. With Perl, setup files are easy because you can write the file in Perl; they simply need to set one or more global variables. To access these variables in a CGI script, first use Perl's require function to import the configuration file. Likewise, if a CGI application grows so large that a single application configuration file is difficult to manage, you can break it into smaller files and have the primary configuration file require these smaller sections.

- **Separating Display from Your Primary Code :** The display associated with a CGI script is one of the most likely things to change in the lifetime of an application. Most Web sites undergo some look and feel change during their evolution, and an application that will be used across several web sites needs to be flexible enough to accommodate all of their individual cosmetic guidelines.

Keeping HTML separate from code so that HTML maintainers have an easier time, it is a good idea to develop the code that handles display separated from the rest of your program logic. This allows you to change the solution you use for generating display with as little effort as possible.

- **Number of Scripts per Application :** CGI applications often consist of many different tasks that must work together. For example, in a basic online store you will have code to display a product catalog, code to update a shopping cart, code to

display the shopping cart, and code to accept and process payment information. Some CGI developers would argue that all of this should be managed by a single CGI script, possibly breaking some functionality out into modules that can be called by this script. Others would argue that a separate CGI scripts should support each page or functional group of pages, possibly moving common code into modules that can be shared by this script.

- **Separating Storage from Your Primary Code :** Separating the code that is responsible for data storage from your core program logic is good architectural design. The manner of storing and retrieving data is a key architecture decision that every application encounters. A simple shopping cart might start out using flat text files to store shopping cart data throughout the user's shopping experience. For this we use relational database such as Oracle or MY SQL or DBM hash files.

- **Using Submit Buttons to Control Flow :** In situations where one form may allow the user to choose different actions, CGI script can take a action by looking at the name of submit button that was chosen. The name and value of submit buttons is only included within form query requests if they were clicked by the user. Thus, you can have multiple submit buttons on the HTML form with different names indicating different paths of logic that the program should follows.

8.8.2 Coding Guidelines

Programmers develop their own style for writing code. This is fine so long as the developer works alone. However, when multiple developers each attempt to impose their own style on a project, it will lead to problems. Code that does not follow one consistent style is much more difficult to read and maintain than uniform code. Thus, if you have more than one developer working on the same project, you should agree on a common style for writing code. Even if you are working alone, it is a good idea to look at common standards so that your style does not become so different that you have problems adapting when you do work with others.

Coding Guidelines include the following topics:

- **Flags and Pragmas**

This covers the first couple of lines of your code:

```
#!/usr/bin/perl -wT
use strict;
```

You may want to require taint mode on all your scripts or allow certain exceptions. You may want to enable warnings by default for all of your scripts too. It is certainly a good idea to require that all scripts use strict and minimize the use of global variables.

- **Capitalization**

This includes the capitalization of

- the variables (both local and global)
- the subroutines
- the modules
- the filenames

The most common convention in Perl is to use lowercase for local variables, subroutines, and filenames; words should be separated by an underscore. Global variables should be capitalized to make them apparent. Module names typically use mixed case without underscores.

- **Bracket placement :** When creating the body of a subroutine, loops, or conditionals, the opening brace can go at the end of the statement preceding it or on the following line. For example, you can declare a subroutine this way:

```
sub sum {
return $_[0] + $_[1];
}
or you could declare it this way:
sub sum
{
return $_[0] + $_[1];
}
```

- **Documentation :** Documentation can include comments within your code adding explanation to sections of code. Documentation can also include an overview of the purpose of a file and how it fits into the larger project. Finally, a project itself may have goals and details that don't fit within particular files but must be captured at a more general level.

- **Whitespace :** Using whitespaces contribute to making code easier to read and thus maintain is an effective use of whitespace. Separate items in lists with spaces, including parameters passed to functions. Include spaces around operators, including parentheses. Line up similar commands on adjacent lines if it helps make the code clear

SHORT QUESTION ANSWERS

Q.1 What are different data access method in CGI scripts?

Ans. The two methods used to access the data are:

1. POST - Sends data to the standard input. The variable CONTENT_LENGTH must be used to tell how much data to read.

2. GET - Sends data to the environment variable QUERY_STRING. The length of this string is limited so it is better to use POST when large amounts of data are expected.

There are many environment variables used by the HTTP to CGI interface which provide additional information about the HTTP transfer.

Q.2 Write a note on Environment variables.

Ans. In CGI, the server prepares the environment variables before it launches the CGI script. These represent the current state of the server that is asking for the

information. The environment variables are not set from the command line but are created on the fly, and lasts only until that particular script is finished. Each script gets its own unique set of variables and multiple scripts can be executed at once, each in its own environment.

Q.2 How data is to be retrieved from the client ?

Ans. HTML forms are the user interface that provides input to your CGI scripts. They are primarily used for two purposes: collecting data and accepting commands. Examples of data you collect may include registration information, payment information, and online surveys. You may also collect commands via forms, such as using menus, checkboxes, lists, and buttons to control various aspects of your application. In many cases, your forms will include elements for both: collecting data as well as application control.

A great advantage of HTML forms is that you can use them to create a frontend for numerous gateways (such as databases or other information servers) that can be accessed by any client without worrying about platform dependency.

```
<FORM ACTION="/cgi-bin/program.pl" METHOD="POST">
```

There are two methods for sending form data: GET and POST. The main difference between these methods is the way in which the form data is passed to the CGI program. If the GET method is used, the query string is simply appended to the URL of the program when the client issues the request to the server. This query string can then be accessed by using the environment variable QUERY_STRING. Here is a sample GET request by the client, which corresponds to the first form example:

```
GET /cgi-bin/program.pl?user=Larry%20Bird&age=35&pass=testing HTTP/1.0
Accept: www/source
Accept: text/html
Accept: text/plain
User-Agent: Lynx/2.4 libwww/2.14
```

Q.3 How data is to be decoded from the input ?

Ans. The algorithm for decoding form data follows:

1) Determine request protocol (either GET or POST) by checking the REQUEST_METHOD environment variable.
2) If the protocol is GET, read the query string from QUERY_STRING and/or the extra path information from PATH_INFO.
3) If the protocol is POST, determine the size of the request using CONTENT_LENGTH and read that amount of data from the standard input.
4) Split the query string on the "&" character, which separates key-value pairs (the format is key=value&key=value...).
5) Decode the hexadecimal and "+" characters in each key-value pair.
6) Create a key-value table with the key as the index.

Q.4 Explain CGI output in brief.

Ans. Script output: The script sends its output to stdout. This output can either be a document generated by the script, or instructions to the server for retrieving the desired output and consist of script naming conventions as HTTP/1.0.

Parsed headers : The output of scripts begins with a small header. This header consists of text lines, in the same format as an HTTP header, terminated by a blank line (a line with only a linefeed or CR/LF).

Content-type : This is the MIME type of the document you are returning.

Location : This is used to specify to the server that you are returning a reference to a document rather than an actual document.

Status : This is used to give the server an HTTP/1.0 status line to send to the client. The format is nnn xxxxx, where nnn is the 3-digit status code, and xxxxx is the reason string, such as "Forbidden".

Q. 5 Describe Architectural guidelines & Coding guidelines.

Ans. CGI developers set guidelines that help their code. In a corporate setting, these guidelines tend to become the standards through which teams of developers understand how to easily read the code that their neighbors produce. There are two types of guidelines:

Architectural Guidelines: Architectural Guidelines include the following tips on how to architect a CGI application.

- Plan for Future Growth
- Use Directories to Organize Your Projects
- Use Relative URLs
- Separate Configuration from Your Primary Code
- Separating Display from Your Primary Code
- Separating Storage from Your Primary Code
- Using Submit Buttons to Control Flow

Coding Guidelines: Programmers develop their own style for writing code. Coding Guidelines include the following topics:

- Flags and Pragmas
- Capitalization
- Bracket placement
- Documentation
- Whitespace

Q.6 List the advantages of CGI scripting?

Ans. Following are the advantages of CGI:

1. CGI programs are relatively safe to run.
2. A CGI program can crash without damaging the server, since it only has limited access to the server.

3. Reduces the burden of server.

a) Sends prepared messages / mails e customer reply

b) Capability to process forms and prepares output based on form input.

c) Hit counts / Page counters.

Q.7 What are STDIN and STDOUT?

Ans. These are mnemonics for standard input and standard output, two predefined stream file handles. Each process already inherits these two handles already open. From the script's point of view, STDIN is what comes from the browser via the server when the post method is used, and the STDOUT is where it writes its output back to the browser. The script picks up the environment variables and reads STDIN as appropriate. It then does whatever it was designed to do and writes its output to STDOUT.

MULTIPLE CHOICE QUESTIONS

1. The CGI programs are the files with extensions _____ or _____

a) CGI, .PL b) CGI, .BL

c) CGI, .DL d) CGI, PL

2. Earlier, servers that were supporting CGI had provided a common directory named _____ to store CGI program

a) cgi-debug b) cgi-bin

c) cgi-jdk d) cgi-jdbc

3. _____environment variable specifies the name of the web server host:

a) SERVER_NAME b) WEB_SERVER

c) WEB_BROWSER d) ALL OF THESE

4. CGI complete its program processing it passes the resultant data to the ––––––––––;

a) Web browser b) Web server

c) Server d) All of these

5. The web server then responds back to the _____ accordingly:

a) Web browser b) Web server

c) Server d) All of these

6. The data can be passed to CGI program in one of the following way:

a) Argument of the CGI program b) Environment variable

c) Standard input d) All of these

7. This information is sent to the CGI program in the _____ environment program:

a) Query_String b) QUERY_STRING

c) QUERY STRING d) ALL OF THESE

8. It specifies the name of user issuing request:

a) REMOTE_USER b) REMOTE USER

c) Remote_user d) None of these

9. It specifies the address of the system of the user issue the request:

a) REMOTE ADDR b) Remote_Addr

c) Remote addr d) REMOTE_ADDR

10. The CGI program use _____ environment variable to read the data correctly:

a) CONTENT_LENGTH b) CONTENT LENGTH

c) Content _length d) None of these

11. Which are the header types :

a) Content-type b) Location

c) Status d) All of these

12. Advantages of CGI:

a) Platform independent b) Language independence

c) Simplicity d) All of these

13. CGI program also known as _____

a) CGI scripts b) Server side web programs

c) Both of these d) None of these

KEY TO OBJECTIVE QUESTIONS

1.a	2.b	3.a	4.b	5.a	6.d	7.b
8.a	9.d	10.a	11.d	12.d	13.a	

IMPORTANT QUESTIONS

Q.1 Explain CGI and steps to establish CGI environment over a Non-CGI compatible server.

Q.2 Explain CGI Environment. Describe the CGI environment variables in details.

Q.3 How can a Perl program determine the type of browser a web client is using?

Q.4 Describe how input from an HTML form is retrieved in a program?

Q.5 How the forms are processed in PERL using CGI ?

Q.6 How Form data is sent through CGI ?

Q.7 How CGI scripts decode from data.

Q.8 Describe Architectural guidelines & Coding guidelines.

Q.9 Write short notes on CGI and Perl. Why they are popular?

****

JAVA SERVER PAGES

IN THIS CHAPTER, YOU WILL LEARN

- Basics of JSP
- JSP Architecture
- JSP Life Cycle
- JSP Objects and Components,
- JSP: request and response objects
- Retrieving the contents of an HTML format
- Retrieving a query string
- Cookies, creating and Reading Cookies
- Using Application Objects

9.1 JAVASERVER PAGES (JSP) BASICS

JAVASERVER PAGES (JSP) is a technology based on the Java language and capable of returning both static and dynamic content to a client browser. Static content and dynamic content can be intermixed. Static contents are HTML, XML, Text and Dynamic contents are Java code, Displaying properties of JavaBeans, Invoking business logic defined in Custom tags. JSP was developed by Sun Microsystems to allow server side development. JSP files are HTML files with special Tags containing Java source code that provide the dynamic content, most of which start with <% and end with %>.

A JavaServer Pages component is a type of Java Servlet that is designed to fulfill the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands. Using JSP, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically. JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages and sharing information between requests, pages etc.

Java Servlet : A Java program that extends the functionality of a Web server, generating dynamic content and interacting with Web clients using a request-response paradigm.

Static Contents :

- Typically static HTML page
- Same display for everyone

Dynamic contents :

- Contents is dynamically generated based on conditions
- Conditions could be User identity, Time of the day, User entered values through forms and selections.

The following shows the Typical Web Server, different clients containing via the Internet to a Web server.

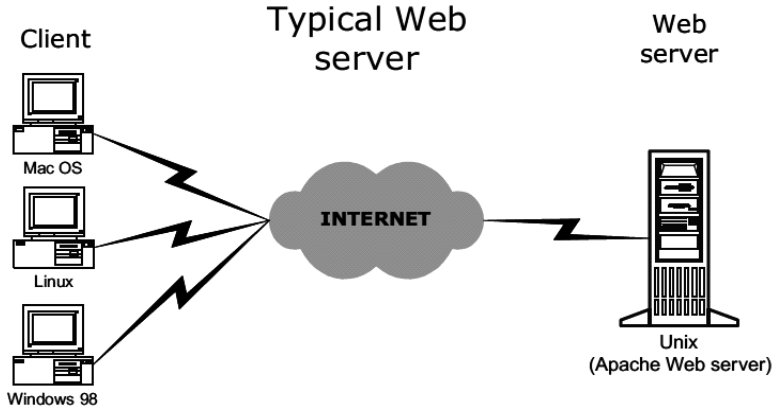


Fig. 9.1

In this example, the Web server is running on Unix and is the very popular Apache Web server. First static web pages were displayed. Typically these were people's first choice experience with making web pages so consisted of My Home Page sites and company marketing information. Afterwards Perl and C were languages used on he web server to provide dynamic contents, soon most languages including Visualbasic, Delphi, C++ and Java could be used to write applications that provide dynamic content using data from text files or data base requests. These were known as CGI server side applications. ASP was develop by Microsoft to allow HTML developers to easily provide dynamic content supported as standard by Microsoft's free Web Server, Internet Information Server (IIS). JSP is the equivalent from Sun Microsystem, a comparison of Asp and JSP will be presented in the following section

The following diagram shows a web server that supports JSP files. Notice that the web server also is connected to a database.

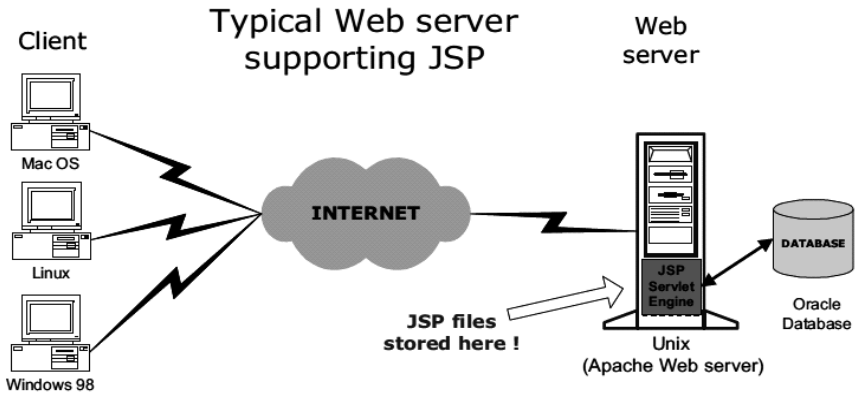


Fig. 9.2

JSP source code runs on the web server in the JSP Servlet Engine. The JSP Servlet engine dynamically generates the HTML and sends the HTML output to the client's web browser.

Why use JSP?

JSP is easy to learn and allows developers to quickly produce we sited and applications in an open and standard way. JSP is based on Java, an object oriented language JSP offers a robust platform for web development.

Main reasons to use JSP:

1. **Separation of dynamic and static content :** This allows for the separation of application logic and Web page design, reducing the complexity of Web site development and making the site easier to maintain.
2. **Platform independence :** Because JSP technology is Java-based, it is platform independent. JSPs can run on any nearly any Web application server. JSPs can be developed on any platform and viewed by any browser because the output of a compiled JSP page is HTML.
3. **Component reuse :** Using JavaBeans and Enterprise JavaBeans, JSPs leverage the inherent reusability offered by these technologies. This enables developers to share components with other developers or their client community, which can speed up Web site development.
4. **Scripting and tags :** JSPs support both embedded JavaScript and tags. JavaScript is typically used to add page-level functionality to the JSP. Tags provide an easy way to embed and modify JavaBean properties and to specify other directives and actions.

You can take one JSP file and move it to another platform, web server or JSP Servlet engine. Moving JSP file from one platform to another.

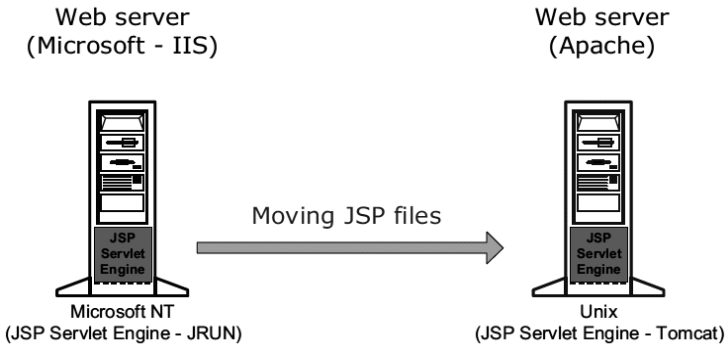


Fig. 9.3

HTML and graphics displayed on the web browser are classed as the presentation layer. The Java code (JSP) on the server is classed as the implementation. By having a separation of presentation and implementation, Web designers work only on the presentation and Java developers concentrate on implementing the application.

Advantages of JSP : Following is the list of other advantages of using JSP over other technologies:

1. **VS. Active Server Pages (ASP):** The advantages of JSP are twofold. First, the dynamic part is written in Java, not Visual Basic or other MS specific language, so it is more powerful and easier to use. Second, it is portable to other operating systems and non-Microsoft Web servers.
2. **VS. Pure Servlets:** It is more convenient to write (and to modify!) regular HTML than to have plenty of println statements that generate the HTML.
3. **VS. Server-Side Includes (SSI):** SSI is really only intended for simple inclusions, not for "real" programs that use form data, make database connections, and the like.
4. **VS. JavaScript:** JavaScript can generate HTML dynamically on the client but can hardly interact with the web server to perform complex tasks like database access and image processing etc.
5. **VS. Static HTML:** Regular HTML, of course, cannot contain dynamic information.

9.2 JSP ARCHITECTURE

JSPs are built on top of SUN's Servlet technology. JSP's are essential an HTML page with special JSP tags embedded. These JSP tags can contain Java code. The JSP extension is .jsp rather than .html or .htm. The JSP Engine parses the .jsp and creates a Java Servlet source file. It then complies the source file into class file, this is done the first time and this way the JSP is probably slower the first time it is accessed. Any time after this the special compiled Servlet is executed and is therefore returns faster.

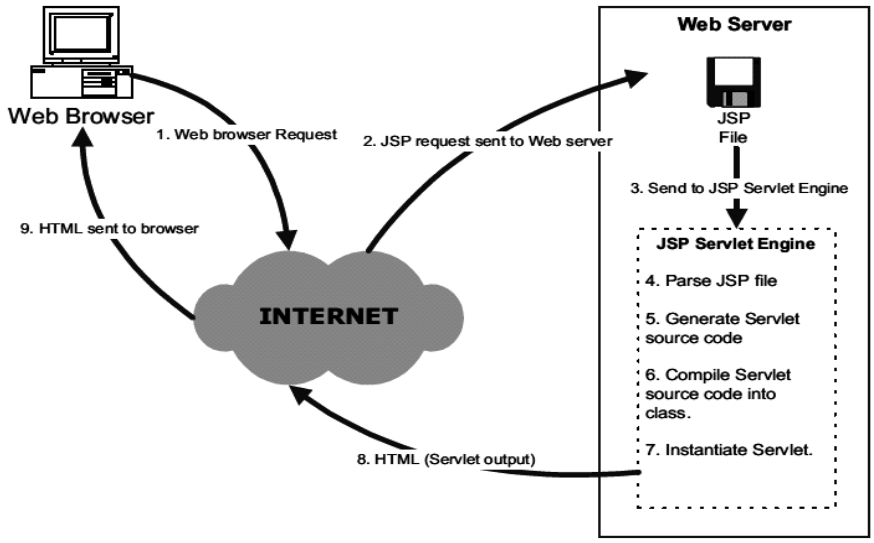


Fig. 9.4

Steps required for a JSP request:

1. The user goes to a web site made using JSP. The user goes to a JSP page (ending with .jsp). the web browser makes the request via the internet.

2. The JSP request gets sent to the Web Server.
3. The Web Server recognizes that the file required is a special (.jsp), therefore passes the JSP file to the JSP Servlet Engine.
4. If the JSP file has been called the first time, the JSP file is parsed, otherwise go to step 7
5. The next step is to generate a special Servlet from the JSP file. All the HTML required is converted to println statements.
6. The Servlet source code is compiled into a class file.
7. The Servlet is instantiated, calling the init and service methods.
8. HTML form the Servlet output sent via the Internet.
9. HTML results are displayed on the user's web browser.

9.3 JSP–LIFE CYCLE

The key to understanding the low-level functionality of JSP is to understand the simple life cycle they follow.

A JSP life cycle can be defined as the entire process from its creation till the destruction which is similar to a servlet life cycle with an additional step which is required to compile a JSP into servlet.

The following are the paths followed by a JSP

- Compilation
- Initialization
- Execution
- Cleanup

The three major phases of JSP life cycle are very similar to Servlet Life Cycle and they are as follows :

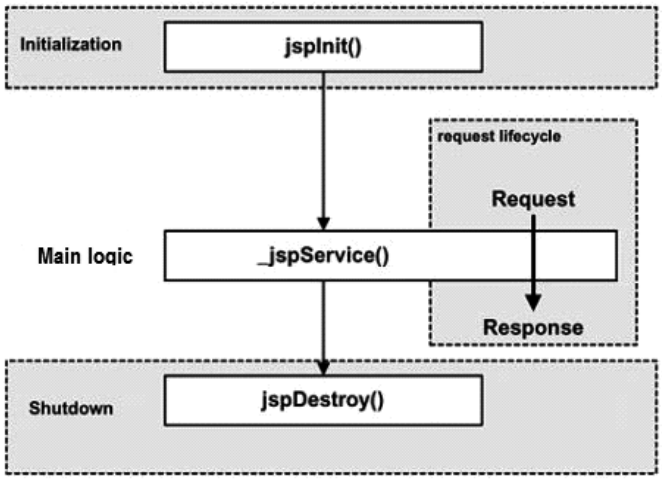


Fig. 9.5

(1) JSP Compilation : When a browser asks for a JSP, the JSP engine first checks to see whether it needs to compile the page. If the page has never been compiled, or if the JSP has been modified since it was last compiled, the JSP engine compiles the page. The compilation process involves three steps:

- Parsing the JSP.
- Turning the JSP into a servlet.
- Compiling the servlet.

(2) JSP Initialization : When a container loads a JSP it invokes the jspInit() method before servicing any requests. If you need to perform JSP-specific initialization, override the jspInit() method:Typically initialization is performed only once and as with the servlet init method, you generally initialize database connections, open files, and create lookup tables in the jspInit method.

```
public void jspInit()
{
   // Initialization code...
}
```

(3) JSP Execution : This phase of the JSP life cycle represents all interactions with requests until the JSP is destroyed.

Whenever a browser requests a JSP and the page has been loaded and initialized, the JSP engine invokes the _jspService() method in the JSP. The **_jspService()** method takes an **HttpServletRequest** and an **HttpServletResponse** as its parameters as follows:

```
void _jspService(HttpServletRequest request, HttpServletResponse response)
{
   // Service handling code...
}
```

The _jspService() method of a JSP is invoked once per a request and is responsible for generating the response for that request and this method is also responsible for generating responses to all seven of the HTTP methods ie. GET, POST, DELETE etc.

4) JSP Cleanup : The destruction phase of the JSP life cycle represents when a JSP is being removed from use by a container.

The **jspDestroy()** method is the JSP equivalent of the destroy method for servlets. Override jspDestroy when you need to perform any cleanup, such as releasing database connections or closing open files.

The jspDestroy() method has the following form:

```
public void jspDestroy()
{
   // Your cleanup code goes here.
}
```

9.4 JSP OBJECTS AND COMPONENTS

- **Scripting Elements:**JSPs are built using ***comments, declarations, expressions and scriptlets***. These are the basic building blocks of JSPs.
- **Directives:**The three directives, page, include, and taglib, extend the basic scripting capabilities by providing compiler directives, including class and package libraries, and by importing custom tag libraries. Custom tag libraries provide XML syntax for accessing external Java code. Tag libraries are an important tool for factoring common scripting elements out of JSPs for improved maintainability.
- **Actions :** Actions further extend JSP by providing "forward "and "include" ?ow control, applet plug-ins, and access to JavaBean components.
- **Implicit Objects :** Implicit objects are instances of speci?c Servlet interfaces (e.g., javax.Servlet.http.HttpServletRequest)that are implemented and exposed by the JSP container. These implicit objects can be used in JSP as if they were part of tentative Java language.

9.4.1 Comments

JSP comments can be encapsulated in "<%--" and "--%>" tags, or as regular Java comments inside a code segment, using "<% /**" and "**/ %>" tags. Comments that should be visible in the ?nal client-side code should be encapsulated with standard HTML comment tags: "<!--" and "-->".

```
<%-- This comment will not appear in the HTTP source sent to the client --%>
<% /** This is a regular Java comment inside a code block. It will not be
sent with the html pushed to the client.  **/ %>
<% //this is a single line comment in a java block.
//These comments are also not sent to the client %>
<!-- Standard HTML comment.  Visible when the client chooses to view
source for the page -->
```

9.4.2 DECLARATIONS

JSP declarations are equivalent to member variables and functions of a class. The declared variables and methods are accessible throughout the JSP. These declarations are identi?ed by using <%! %> or the XML equivalent <jsp:declaration></jsp:declaration> tags. You can combine multiple declarations in the same set of tags, including combinations of variables and methods.

```
<%-- Declaring a page level String variable --%>
<%! String bookName = "J2EE CodeNote"; %>
<%-- Declaring a page level public method --%>
<%! public String palindrome(String inString) {
```

```
String outString = "";
for (int i = inString.length(); i > 0; i--) {
outString = outString + inString.charAt(i-1);
}
return outString;
} // palindrome
%>
```

9.4.3 Expressions

An expression is an in-line function that writes text to the buffer. Each set of expression tags encloses a fragment of Java code that must evaluate to a String. Expressions are enclosed in either <%= %> or <jsp:expression> </jsp:expression> tags. The JSP container will generate compilation errors if your expression cannot be converted into a String, or if you have placed a semicolon at the end of your Java statement.

```
<%! String text = "Wow!";%>
This is some html and an expression: <%=text%> </br>
<%-- The same thing using the XML tags--%>
<jsp:declaration>String text = "Wow!";</jsp:declaration> </br>
This is some html and an expression:
<jsp:expression>text</jsp:expression>
<%! String text = "Wow!";%>
This is some html and an expression: <%out.print(text)%> </br>
```

9.4.4 Scripts in JSP

A scriptlet can contain any number of JAVA language statements, variable or method declarations, or expressions that are valid in the page scripting language.

Following is the syntax of Scriptlet :

```
<% code fragment %>
```

You can write **XML equivalent** of the above syntax as follows:

```
<jsp:scriptlet>
code fragment
</jsp:scriptlet>
```

Any text, HTML tags, or JSP elements you write must be outside the scriptlet. Following is the simple and first example for JSP:

```
<html>
<head><title>Hello World</title></head>
<body>
Hello World!<br/>
<%
out.println("Your IP address is " + request.getRemoteAddr());
%>
</body>
</html>
```

NOTE : Assuming that Apache Tomcat is installed in C:\apache-tomcat and your environment is setup as per environment setup tutorial.

Let us keep above code in JSP file hello.jsp and put this file in **C:\apache-tomcat\webapps\ROOT** directory and try to browse it by giving URL http://localhost:8080/hello.jsp. This would generate following result:

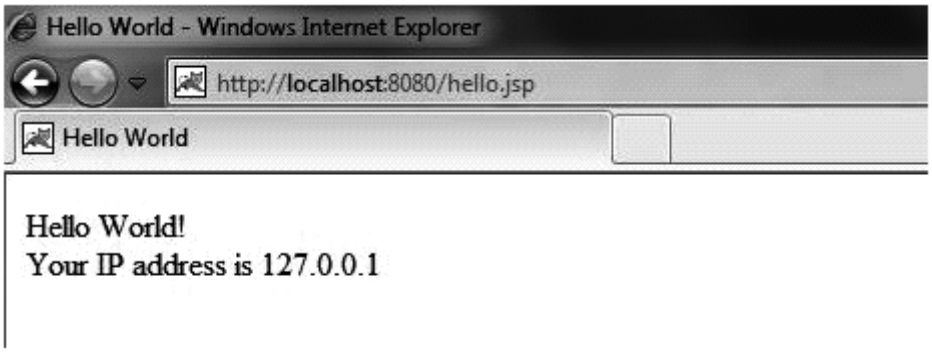


Fig. 9.6

9.4.5 Directives

Directives and actions extend the basic JSP syntax. Directives are instructions from the JSP to the container that can be used to set the page properties, to import Java classes and packages, and to include external web pages and custom tag libraries. The three directives are:

1. ***Page-***The page directive sets the attributes of the page and provides the functionality for importing Java classes.
2. ***Include***-The include directive adds modularity to JSP by allowing you to include the contents of external pages in your JSP.
3. ***Taglib***-The Taglib directive is used for Custom Tag Libraries.

The page Directive : The **page** directive is used to provide instructions to the container that pertain to the current JSP page. You may code page directives anywhere in your JSP page. By convention, page directives are coded at the top of the JSP page.

Following is the basic syntax of page directive:

```
<%@ page attribute="value" %>
```

You can write XML equivalent of the above syntax as follows:

```
<jsp:directive.page attribute="value" />
```

Following is the list of attributes associated with page directive:

Attribute	Purpose
Buffer	Specifies a buffering model for the output stream.
autoFlush	Controls the behavior of the servlet output buffer.
contentType	Defines the character encoding scheme.
errorPage	Defines the URL of another JSP that reports on Java unchecked runtime exceptions.
isErrorPage	Indicates if this JSP page is a URL specified by another JSP page's errorPage attribute.
extends	Specifies a superclass that the generated servlet must extend
import	Specifies a list of packages or classes for use in the JSP as the Java import statement does for Java classes.
info	Defines a string that can be accessed with the servlet's getServletInfo() method.
isThreadSafe	Defines the threading model for the generated servlet.
language	Defines the programming language used in the JSP page.
session	Specifies whether or not the JSP page participates in HTTP sessions
isELIgnored	Specifies whether or not EL expression within the JSP page will be ignored.
isScriptingEnabled	Determines if scripting elements are allowed for use.

The include Directive : The **include** directive is used to includes a file during the translation phase. This directive tells the container to merge the content of other external files with the current JSP during the translation phase. You may code include directives anywhere in your JSP page.

The general usage form of this directive is as follows:

```
<%@ include file="relative url" >
```

The filename in the include directive is actually a relative URL. If you just specify a filename with no associated path, the JSP compiler assumes that the file is in the same directory as your JSP.

You can write XML equivalent of the above syntax as follows:

```
<jsp:directive.include file="relative url" />
```

The taglib Directive: The JavaServer Pages API allows you to define custom JSP tags that look like HTML or XML tags and a tag library is a set of user-defined tags that implement custom behavior. The **taglib** directive declares that your JSP page uses a set of custom tags, identifies the location of the library, and provides a

means for identifying the custom tags in your JSP page.

The taglib directive follows the following syntax:

```
<%@ taglib uri="uri" prefix="prefixOfTag" >
```

Where the **uri** attribute value resolves to a location the container understands and the prefix attribute informs a container what bits of markup are custom actions.

You can write XML equivalent of the above syntax as follows:

```
<jsp:directive.taglib uri="uri" prefix="prefixOfTag" />
```

9.4.6 JSP–Actions

JSP actions use constructs in XML syntax to control the behavior of the servlet engine. You can dynamically insert a file, reuse JavaBeans components, forward the user to another page, or generate HTML for the Java plugin. There is only one syntax for the Action element, as it conforms to the XML standard

```
<jsp:action_name attribute="value" />
```

Action elements are basically predefined functions and there are following JSP actions available:

Syntax	Purpose
jsp:include	Includes a file at the time the page is requested
jsp:useBean	Finds or instantiates a JavaBean
jsp:setProperty	Sets the property of a JavaBean
jsp:getProperty	Inserts the property of a JavaBean into the output
jsp:forward	Forwards the requester to a new page
jsp:element	Defines XML elements dynamically.
Flush	The relative URL of the page to be included. The boolean attribute determines whether the included resource has its buffer flushed before it is included.

The <jsp:include> Action

This action lets you insert files into the page being generated. The syntax looks like this:

```
<jsp:include page="relative URL" flush="true" />
```

Unlike the include directive, which inserts the file at the time the JSP page is translated into a servlet, this action inserts the file at the time the page is requested.

Example:

Let us define following two files (a)date.jps and (b) main.jsp as follows:

Following is the content of date.jsp file:

```
<p>
Today's date: <%= (new java.util.Date()).toLocaleString()%>
</p>
```

Here is the content of main.jsp file:

```
<html>
<head> <title>The include Action Example</title> </head>
<body>
<center>
<h2>The include action Example</h2>
<jsp:include page="date.jsp" flush="true" />
</center>
</body>
</html>
```

Now let us keep all these files in root directory and try to access main.jsp. This would display result something like this:

```
The include action Example
Today's date: 12-Sep-2010 14:54:22
```

9.5 JSP CONFIGURING

JSP needs any web server; this can be tomcat by apache, WebLogic by bea, or WebSphere by IBM. All jsp should be deployed inside web server. We will use Tomcat server to run JSP, this Tomcat server can run on any platform like windows or linux.

Installation of Tomcat on windows or Installation of Tomcat on linux.

After successful installation of tomcat and JSP we need IDE integrated development environment. These IDE provide software development facilities, help lots in programming. This IDE can contain source code editor, debugger, compiler, automatic generation code tools, and GUI view mode tools which show output at a run-time.

We suggest using, Dreamweaver from adobe, or eclipse with myEclipse plugin, NetBeans from sun. Or sun studio creator from sun. These IDEs help in Visual programming

File and folder structure of tomcat

Tomcat
Bin
Conf
Lib
Logs
Tmp
+Webapps

Doc
Example
File
Host-manager
ROOT
jsp
+Work
Catalina

9.5.1 Troubleshooting

Troubleshooting is a form of problem solving most often applied to repair of failed products or processes. It is a logical, systematic search for the source of a problem so that it can be solved, and so the product or process can be made operational again. Troubleshooting is needed to develop and maintain complex systems where the symptoms of a problem can have many possible causes. Troubleshooting is used in many fields such as engineering, system administration, electronics, automotive repair, and diagnostic medicine. Troubleshooting requires identification of the malfunction(s) or symptoms within a system. Then, experience is commonly used to generate possible causes of the symptoms. Determining which cause is most likely is often a process of elimination - eliminating potential causes of a problem. Finally, troubleshooting requires confirmation that the solution restores the product or process to its working state. In general, troubleshooting is the identification of, or diagnosis of "trouble" in a [system] caused by a failure of some kind. The problem is initially described as symptoms of malfunction, and troubleshooting is the process of determining the causes of these symptoms.

A system can be described in terms of its expected, desired or intended behavior (usually, for artificial systems, its purpose). Events or inputs to the system are expected to generate specific results or outputs. (For example selecting the "print" option from various computer applications is intended to result in a hardcopy emerging from some specific device). Any unexpected or undesirable behavior is a symptom. Troubleshooting is the process of isolating the specific cause or causes of the symptom. Frequently the symptom is a failure of the product or process to produce any results. (Nothing was printed, for example).

9.6 IMPLICIT OBJECTS

Implicit Objects in JSP are objects that are automatically available in JSP. Implicit Objects are Java objects that the JSP Container provides to a developer to access them in their program using JavaBeans and Servlets. These objects are called implicit objects because they are automatically instantiated. There are many implicit objects available. The page and con?g objects are used to access the Servlet that results when the container compiles the JSP. These objects are very rarely used because the various scripting elements, directives, and actions already provide the same functionality. The most commonly used objects are request,

response, application, and session objects. There are many implicit objects available. Some of them are:

Request: The class or the interface name of the object request is http.httpservletrequest. The client first makes a request that is then passed to the server. The requested object is used to take the value from client's web browser and pass it to the server. This is performed using HTTP request like headers, cookies and arguments.

Response: This denotes the HTTP Response data. The result or the information from a request is denoted by this object. This is in contrast to the request object. The class or the interface name of the object response is http.HttpServletResponse. Generally, the object response is used with cookies. The response object is also used with HTTP Headers.

Session: This denotes the data associated with a specific session of user. The class or the interface name of the object Session is http.HttpSession. The previous two objects, request and response, are used to pass information from web browser to server and from server to web browser respectively. The Session Object provides the connection or association between the client and the server. The main use of Session Objects is for maintaining states when there are multiple page requests. This will be explained in further detail in following sections.

Out: This denotes the Output stream in the context of page. The class or the interface name of the Out object is jsp.JspWriter. The Out object is written: Javax.servlet.jsp.JspWriter

PageContext: This is used to access page attributes and also to access all the namespaces associated with a JSP page. The class or the interface name of the object PageContext is jsp.pageContext. The object PageContext is written: Javax.servlet.jsp.pagecontext

Application: This is used to share the data with all application pages. The class or the interface name of the Application object is ServletContext. The Application object is written: Javax.servlet.http.ServletContext

Config: This is used to get information regarding the Servlet configuration, stored in the Config object. The class or the interface name of the Config object is ServletConfig. The object Config is written Javax.servlet.http.ServletConfig

Page: The Page object denotes the JSP page, used for calling any instance of a Page's servlet. The class or the interface name of the Page object is jsp.HttpJspPage. The Page object is written: Java.lang.Object

The most commonly used implicit objects are request, response and session objects.

For example, to store attributes in a session object, you would use code like this:

```
<%session.setAttribute("number", new Float(42.5));%>
```

To retrieve the parameters from the request object, you would use code like this:

```
<%@ page import="java.util.*" %>
```

```
<% Enumeration params = request.getParameterNames();%>
```

9.7 JSP REQUEST OBJECTS

The request object in JSP is used to get the values that the client passes to the web server during an HTTP request. The request object is used to take the value from the client's web browser and pass it to the server. This is performed using an HTTP request such as: headers, cookies or arguments. The class or the interface name of the object request is http.httpservletrequest.

The object request is written: Javax.servlet.http.httpservletrequest.

Methods of request Object : There are numerous methods available for request object. Some of them are:

```
getCookies()
getHeader(String name)
getHeaderNames()
getAttribute(String name)
getAttributeNames()
getMethod()
getParameter(String name)
getCookies()
```

The getCookies() : method of request object returns all cookies sent with the request information by the client. The cookies are returned as an array of Cookie Objects. We will see in detail about JSP cookies in the coming sections.

9.8 JSP RESPONSE OBJECTS

The response object denotes the HTTP Response data. The result or the information of a request is denoted with this object. The response object handles the output of the client. This contrasts with the request object. The class or the interface name of the response object is http.HttpServletResponse.

1. The response object is written: Javax.servlet.http.httpservletresponse.
2. The response object is generally used by cookies.
3. The response object is also used with HTTP Headers.

There are numerous methods available for response object. Some of them are:

```
setContentType()
addCookie(Cookie cookie)
addHeader(String name, String value)
containsHeader(String name)
setHeader(String name, String value)
sendRedirect(String)
sendError(int status_code)
```

setContentType()

setContentType() method of response object is used to set the MIME type and character encoding for the page.

General syntax of setContentType() of response object is as follows:

```
response.setContentType();
```

For example:

```
response.setContentType("text/html");
```

The above statement is used to set the content type as text/html dynamically.

9.9 RETRIEVING THE CONTENTS OF A HTML FORM

Forms are, of course, the most important way of getting information from the customer of a web site. In this section, we'll just create a simple color survey and print the results back to the user. First, create the entry form. Our HTML form will send its answers to form.jsp for processing. For this example, the name="name" and name="color" are very important. You will use these keys to extract the user's responses.

form.html

```
<form action="form.jsp" method="get">
<table><tr><td><b>Name</b> <td><input type="text" name="name">
<tr><td><b>Favorite color</b> <td><input type="text" name="color"> </table>
<input type="submit" value="Send">
</form>
```

Keeps the browser request information in the request object. The request object contains the environment variables you may be familiar with from CGI programming. For example, it has the browser type, any HTTP headers, the server name and the browser IP address. You can get form values using request.getParameter object. The following JSP script will extract the form values and print them right back to the user.

form.jsp

```
Name: <%= request.getParameter("name") %> <br>
Color: <%= request.getParameter("color") %>
```

9.10 RETRIEVING A QUERY STRING

An include action executes the included JSP page and appends the generated output onto its own output stream. Request parameters parsed from the URL's query string are available not only to the main JSP page but to all included JSP pages as well. It is possible to temporarily override a request parameter or to temporarily introduce a new request parameter when calling a JSP page. This is done by using the jsp:param action.

In this example, param1 is specified in the query string and is automatically made available to the callee JSP page. param2 is also specified in the query string but is overridden by the caller. Notice that param2 reverts to its original value after the call. param3 is a new request parameter created by the caller. Notice that param3 is only available to the callee and when the callee returns, param3 no longer exists. Here is the caller JSP page:

```
<html>
<body>
<jsp:include page="callee.jsp" />
    <jsp:param name="param2" value="value2" />
    <jsp:param name="param3" value="value3" />
</jsp:include>
Caller
param1: <%=request.getParameter( "param1") %>
param2: <%=request.getParameter( "param2") %>
param3: <%=request.getParameter( "param3") %>
</body>
</html>
Here is the JSP page being called:
Callee:
param1: <%=request.getParameter( "param1") %>
param2: <%=request.getParameter( "param2") %>
param3: <%=request.getParameter( "param3") %>
```

the output would be:

```
Callee:
param1: a
param2: value2
param3: value3


Caller:
param1: a
param2: b
param3: null
```

If the example is called with the URL:

http://hostname.com?param1=a&param2=b

9.11 WORKING WITH BEANS

Java Beans are reusable components. They are used to separate Business logic from the Presentation logic. Internally, a bean is just an instance of a class.

JSP's provide three basic tags for working with Beans.

```
<jsp:useBean id="bean name" class="bean class" scope = "page | request
| session |application"/>
```

bean name = the name that refers to the bean.

Bean class = name of the java class that defines the bean.

```
<jsp:setProperty name = "id" property = "someProperty" value = "someValue"/>
```

id = the name of the bean as specified in the useBean tag.

property = name of the property to be passed to the bean.

value = value of that particular property .

An variant for this tag is the property attribute can be replaced by an " * ". What this does is that it accepts all the form parameters and thus reduces the need for writing multiple setProperty tags. The only consideration is that the form parameter names should be the same as that of the bean property names.

```
<jsp:getProperty name = "id" property = "someProperty"/>
```

Here the property is the name of the property whose value is to be obtained from the bean.

9.11.1 Bean Scopes

These define the range and lifespan of the bean.

The different options are:

1. **Page scope :** Any object whose scope is the page will disappear as soon as the current page finishes generating. The object with a page scope may be modified as often as desired within the particular page but the changes are lost as soon as the page exists. By default all beans have page scope.

2. **Request scope :** Any objects created in the request scope will be available as long as the request object is. For example if the JSP page uses an jsp:forward tag, then the bean should be applicable in the forwarded JSP also, if the scope defined is of Request scope.

3. **The Session scope :** In JSP terms, the data associated with the user has session scope. A session does not correspond directly to the user; rather, it corresponds with a particular period of time the user spends at a site. Typically, this period is defined as all the visits a user makes to a site between starting and existing his browser.

9.11.2 The Bean Structure

The most basic kind of bean simply exposes a number of properties by following a few simple rules regarding method names. The Java BEAN is not much different from an java program. The main differences are the signature methods being used in a bean. For passing parameters to a bean, there has to be a corresponding get/set method for every parameter. Together these methods are known as accessors. For example:

Suppose we want to pass a parameter "name" to the bean and then return it in the capital form. In the bean, there has to be an setName() method and an corresponding getProperty() method. A point to be noted is that the first letter of the property name is capitalized.(Here, N is in capital). Also, it is possible to have either get or set in a bean, depending on the requirement for a read only or a write only property.

An example for a Database connection bean is as shown:

```
package SQLBean;
import java.sql.*;
import java.io.*;
public class DbBean {
    String dbURL = "jdbc:db2:sample";
String dbDriver = "COM.ibm.db2.jdbc.app.DB2Driver";
private Connection dbCon;
    public DbBean(){
super();
}
    public boolean connect() throws ClassNotFoundException,SQLException{
Class.forName(dbDriver);
dbCon = DriverManager.getConnection(dbURL);
return true;
}
    public void close() throws SQLException{
dbCon.close();
}
    public ResultSet execSQL(String sql) throws SQLException{
Statement s = dbCon.createStatement();
ResultSet r = s.executeQuery(sql);
return (r == null) ? null : r;
}
public int updateSQL(String sql) throws SQLException{
Statement s = dbCon.createStatement();
int r = s.executeUpdate(sql);
return (r == 0) ? 0 : r;
}
}
```

The description is as follows: This bean is packaged in a folder called as "SQLBean". The name of the class file of the bean is DbBean. For this bean we have hardcoded the Database Driver and the URL. All the statements such as connecting to the database, fetching the driver etc are encapsulated in the bean.

9.12 COOKIES

Cookies are short pieces of data sent by web servers to the client browser. The cookies are saved to clients hard disk in the form of small text file. Cookies helps the web servers to identify web users, by this way server tracks the user. Cookies pay very important role in the session tracking.

Cookie Class **:** In JSP cookies are the object of the class javax.servlet.http.Cookie. This class is used to creates a cookie, a small amount of information sent by a servlet to a Web browser, saved by the browser, and later sent back to the server.

A cookie's value can uniquely identify a client, so cookies are commonly used for session management. A cookie has a name, a single value, and optional attributes such as a comment, path and domain qualifiers, a maximum age, and a version number. There are three steps involved in identifying returning users:

1. Server script sends a set of cookies to the browser. For example name, age, or identification number etc.
2. Browser stores this information on local machine for future use.
3. When next time browser sends any request to web server then it sends those cookies information to the server and server uses that information to identify the user or may be for some other purpose as well.

The Anatomy of a Cookie **:** Cookies are usually set in an HTTP header (although JavaScript can also set a cookie directly on a browser). A JSP that sets a cookie might send headers that look something like this:

```
HTTP/1.1 200 OK
Date: Fri, 04 Feb 2000 21:03:38 GMT
Server: Apache/1.3.9 (UNIX) PHP/4.0b3
Set-Cookie: name=xyz; expires=Friday, 04-Feb-07 22:03:38 GMT;
path=/; domain=tutorialspoint.com
Connection: close
Content-Type: text/html
```

As you can see, the Set-Cookie header contains a name value pair, a GMT date, a path and a domain. The name and value will be URL encoded. The expire field is an instruction to the browser to "forget" the cookie after the given time and date. If the browser is configured to store cookies, it will then keep this information until the expiry date. If the user points the browser at any page that matches the path and domain of the cookie, it will resend the cookie to the server. The browser's headers might look something like this:

```
GET / HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.6 (X11; I; Linux 2.2.6-15apmac ppc)
Host: zink.demon.co.uk:1126
Accept: image/gif, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
Cookie: name=xyz
```

9.12.1 Setting Cookies with JSP

Setting cookies with JSP involves three steps:

1. Creating a Cookie object: You call the Cookie constructor with a cookie name and a cookie value, both of which are strings.

```
Cookie cookie = new Cookie("key","value");
```

2. Setting the maximum age: You use setMaxAge to specify how long (in seconds) the cookie should be valid. Following would set up a cookie for 24 hours.

```
cookie.setMaxAge(60*60*24);
```

3. Sending the Cookie into the HTTP response headers: You use response.addCookie to add cookies in the HTTP response header as follows:

```
response.addCookie(cookie);
```

Example:

Let us take a Form Example to set the cookies for first and last name.

```
<%
// Create cookies for first and last names.
Cookie firstName = new Cookie("first_name",
    request.getParameter("first_name"));
Cookie lastName = new Cookie("last_name",
    request.getParameter("last_name"));
// Set expiry date after 24 Hrs for both the cookies.
firstName.setMaxAge(60*60*24);
lastName.setMaxAge(60*60*24);
// Add both the cookies in the response header.
response.addCookie( firstName );
response.addCookie( lastName );
%>
<html>
<head>
<title>Setting Cookies</title>
</head>
<body>
<center>
<h1>Setting Cookies</h1>
</center>
<ul>
<li><p><b>First Name:</b>
<%= request.getParameter("first_name")%>
</p></li>
<li><p><b>Last  Name:</b>
<%= request.getParameter("last_name")%>
</p></li>
</ul>
</body>
</html>
```

Let us put above code in main.jsp file and use it in the following HTML page:

```
<html>
<body>
<form action="main.jsp" method="GET">
First Name: <input type="text" name="first_name">
Last Name: <input type="text" name="last_name" />
<input type="submit" value="Submit" />
</form>
</body> </html>
```

Keep above HTML content in a file hello.jsp and put hello.jsp and main.jsp in <Tomcat-installation-directory>/webapps/ROOT directory. When you would access http://localhost:8080/hello.jsp, here is the actual output of the above form.

Fig. 9.7

Try to enter First Name and Last Name and then click submit button. This would display first name and last name on your screen and same time it would set two cookies firstName and lastName which would be passed back to the server when next time you would press Submit button. Next section would explain you how you would access these cookies back in your web application.

9.12.2 Reading Cookies with JSP

To read cookies, you need to create an array of javax.servlet.http.Cookie objects by calling the ***getCookies( )*** method of *HttpServletRequest*. Then cycle through the array, and use getName() and getValue() methods to access each cookie and associated value.

Example:

Let us read cookies which we have set in previous example:

```
<html>
<head> <title>Reading Cookies</title> </head>
<body>
<center> <h1>Reading Cookies</h1> </center>
<%
    Cookie cookie = null;
    Cookie[] cookies = null;   // Get an array of Cookies associated with
this domain
    cookies = request.getCookies();
    if( cookies != null ){
        out.println("<h2> Found Cookies Name and Value</h2>");
        for (int i = 0; i < cookies.length; i++){
```

```
            cookie = cookies[i];
            out.print("Name : " + cookie.getName( ) + ",  ");
            out.print("Value: " + cookie.getValue( )+" <br/>");
        }
    } else {       out.println("<h2>No cookies founds</h2>");   }
%>
</body>
</html>
```

Now let us put above code in main.jsp file and try to access it. If you would have set first_name cookie as "John" and last_name cookie as "Player" then running http://localhost:8080/main.jsp would display the following result:

Found Cookies Name and Value

Name : first_name, Value: John

Name : last_name, Value: Player

9.13 JSP Application Objects

Application Object is used to share the data with all application pages. Thus, all users share information of a given application using the Application object. The Application object is accessed by any JSP present in the application. The class or the interface name of the object application is ServletContext. The application object is written as:

```
Javax.servlet.http.ServletContext
```

Methods of Application Object : There are numerous methods available for Application object. Some of the methods of Application object are:

- getAttribute(String name)
- getAttributeNames
- setAttribute(String objName, Object object)
- removeAttribute(String objName)
- getServerInfo()
- getInitParameter(String name)

getInitParameterNames getAttribute(String name)

The method getAttribute of Application object is used to return the attribute with the specified name. It returns the object given in parameter with name. If the object with name given in parameter of this getAttribute does not exist, then null value is returned. General syntax of getAttribute method of Application object is as follows: For example–

```
application.getAttribute("Web Engineering");
```

The above statement returns the object Web Engineering.

SHORT QUESTION ANSWERS

Q. 1 What is JSP?

Ans. JSP stands for Java Server Pages. JSP is a server-side technology Java Server Pages are an extension to the Java Servlet technology that was developed by Sun.

JSPs have dynamic scripting capability that works in tandem with HTML code, separating the page logic from the static elements -- the actual design and display of the page -- to help make the HTML more functional (i.e. dynamic database queries). JSPs are not restricted to any specific platform or server. It was originally created as an alternative to Microsoft's ASPs (Active Server Pages). Recently, however, Microsoft has countered JSP technology with its own ASP.NET, part of the .NET initiative.

Q. 2 What are the advantages of JSP over Servlet?

Ans. JSP is a serverside technology to make content generation a simple appear.The advantage of JSP is that they are document-centric. Servlets, on the other hand, look and act like programs. A Java Server Page can contain Java program fragments that instantiate and execute Java classes, but these occur inside an HTML template file and are primarily used to generate dynamic content. Some of the JSP functionality can be achieved on the client, using JavaScript. The power of JSP is that it is server-based and provides a framework for Web application development.

Q.3 What is the life-cycle of JSP?

Ans. When a request is mapped to a JSP page for the first time, it translates the JSP page into a servlet class and compiles the class. It is this servlet that services the client requests. A JSP page has seven phases in its lifecycle, as listed below in the sequence of occurrence:

1. Translation
2. Compilation
3. Loading the class
4. Instantiating the class
5. jspInit() invocation
6. jspService() invocation
7. jspDestroy() invocation

Q. 4 What are implicit objects in JSP?

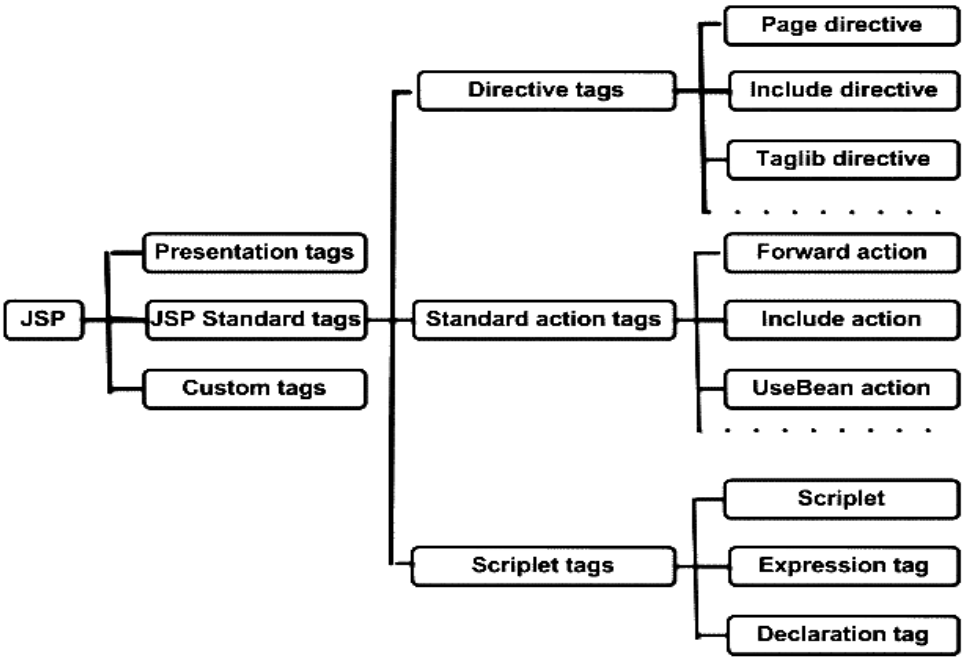
Ans. Implicit objects in JSP are the Java objects that the JSP Container makes available to developers in each page. These objects need not be declared or instantiated by the JSP author. They are automatically instantiated by the container and are accessed using standard variables; hence, they are called implicit objects.The implicit objects available in JSP are as follows:

1. request
2. response
3. pageContext
4. session
5. application
6. out
7. config
8. page
9. exception

The implicit objects are parsed by the container and inserted into the generated servlet code. They are available only within the jspService method and not in any declaration.

Q. 5 What are the different types of JSP tags?

Ans. The different types of JSP tags are as follows:



Q. 6 What are JSP directives?

Ans. JSP directives are messages for the JSP engine. i.e., JSP directives serve as a message from a JSP page to the JSP container and control the processing of the entire page

1. They are used to set global values such as a class declaration, method implementation, output content type, etc.
2. They do not produce any output to the client.
3. Directives are always enclosed within <%@ ….. %> tag.
4. Ex: page directive, include directive, etc.

Q. 7 What is page directive?

Ans. A page directive is to inform the JSP engine about the headers or facilities that page should get from the environment.

1. Typically, the page directive is found at the top of almost all of our JSP pages.
2. There can be any number of page directives within a JSP page (although the attribute - value pair must be unique).
3. The syntax of the include directive is: <%@ page attribute="value">
4. Example: <%@ include file="header.jsp" %>

Q. 8 What are the attributes of page directive?

Ans. There are thirteen attributes defined for a page directive of which the important attributes are as follows:

1. **import :** It specifies the packages that are to be imported.
2. **session:** It specifies whether a session data is available to the JSP page.
3. **Content Type:** It allows a user to set the content-type for a page.
4. **isEL Ignored:** It specifies whether the EL expressions are ignored when a JSP is translated to a servlet.

Q.9 What is the include directive?

Ans. There are thirteen attributes defined for a include directive of which the important attributes are as follows:

1. The include directive is used to statically insert the contents of a resource into the current JSP.
2. This enables a user to reuse the code without duplicating it, and includes the contents of the specified file at the translation time.
3. The syntax of the include directive is as follows:

   <%@ include file = "FileName" %>
4. This directive has only one attribute called file that specifies the name of the file to be included.

Q.10 What are scripting elements?

Ans. JSP scripting elements let you insert Java code into the servlet that will be generated from the current JSP page. There are three forms:

1. Expressions of the form <%= expression %> that are evaluated and inserted into the output,
2. Scriptlets of the form <% code %> that are inserted into the servlet's service method,
3. Declarations of the form <%! code %> that are inserted into the body of the servlet class, outside of any existing methods.

Q.11 What is cookies? Explain its importance.

Ans. A cookie is a small file that a Web server can store on your machine. Its purpose is to allow a Web server to personalize a Web page, depending on whether you have been to that Web site before, and what you may have told it during previous sessions. When you return to that Web site in the future the Web server can read its cookie, recall this information, and structure its Web pages accordingly. However, cookies do make it easier for advertising companies to gather information about your browsing habits.

Q.12 What are different methods to add and read cookies ?

Ans. Cookie is a small bit of text information which a web server sends to a browser

and which browsers returns the cookie when it visits the same site again. In cookie the information is stored in the following form - a name, a value pair.

Add Cookie to response object:

```
Cookie cookie = new Cookie ("name",value);
cookie.setPath("/");
cookie.setDomain(DOMAIN_NAME); DOMAIN_NAME may be .techfaq360.com
cookie.setMaxAge(2* 7 * 24 * 60 * 60);// 2 week
response.addCookie(cookie);
```

Get cookie from requested Object :

```
Cookie myCookie = null;
Cookie cookies [] = request.getCookies ();
if (cookies != null)
for (int i = 0; i < cookies.length; i++)
{
if (cookies [i].getName().equals ("name")) // the name of the cookie
you have added
{
myCookie = cookies[i];
break;
} }
```

MULTIPLE CHOICE QUESTIONS

1. A _ _ _ _ _ is a small amount of data that is saved on the clients machine and can be created by and referenced by a Java Servlet

a. servlet b. cookie

c. session d. bookie

2. HTTP is _ _ _ _ protocol.

a. state b. stateless

c. bound d. unbound

3. WML stands for.

a. Web Markup Language b. Wrapper Markup Language

c. Wireless Markup Language d. Web page Markup Language

4. JSP pages are _ _ _ _ for efficient server processing.

a. Recompiled b. Compiled

c. Executed d. Re executed

5. JSP is an integral part of _ _ _ _ _ _ _ _.

a. J2ME b. J2SE

c. J2EE d. J2CS

6. JSP stands for

a. Java Script Pages b. Java Server Pages

c. Java Servlet pages d. Java Support Pages

7. The problem with servlets is

a. Processing the request and generating the response are both handled by a single servlet class

b. Processing the request and generating the response are both handled by a two servlet classes

c. Processing the request and generating the response are both handled by a three servlet classes

d. Processing the request and generating the response are both handled by a four servlet classes

8. The anatomy of a JSP page contains

a. JSP container and JSP request b. JSP request and template text

c. JSP element and template text d. Template text and JSP request

9. Which of the following is not a popular technology for developing dynamic web sites?

a. HPH b. ASP c. JSP d. PHP

10. Directive elements are defined in _ _ _ _ _ _ tag.

a. <%.....%> b. <%@.....%>

c. <%=.....%> d. <%!.....%>

11. A servlet container and a JSP container are often combined in one package under the name _ _ _ _ _.

a. Server b. JSTL

c. JSP Container d. Web Container

12. The _ _ _ _ _ _ elements, specify information about the page itself that remains the same between requests.

a. standard action b. custom action

c. scripting d. directive

13. The _ _ _ _ _ _ is also responsible for invoking the JSP page implementation class to process each request and generate the response.

a. JSP container b. Servlet container

c. Web container d. Server container

14. JRE stands for

a. Java Runtime Environment b. Jakartha Runtime Environment

c. Java Recovered Exception d. Java Runtime Exception

15. Tomcat is pure

a. web server b. web container

c. web browser d. web application

16. Which protocol we will use in web applications?

a. FTP b. SMTP

c. HTTP d. TCP/IP

KEY TO OBJECTIVE QUESTIONS

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

IMPORTANT QUESTIONS

Q.1 What is JSP ? Explain its architecture in detail.

Q.2 Explain in detail the life cycle of JSP .

Q.3 What is the difference between JSP and Servlets ?

Q.4 What are the advantages of JSP over Servlet?

Q.5 Differentiate between static and dynamic web pages. How does JSP help creating dynamic web pages? Explain the JSP request and response object using suitable examples.

Q.6 Write short note on JSP objects & Components.

Q.7 Explain JSP Expression, JSP Declaration, JSP Scriptlet?

Q.8 Explain different JSP directives and components.

Q.5 Write down the various attributes for the page directives in JSP.

Q.9 What is the difference between include directive and include action ?

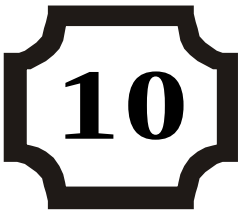
Q.10 What are implicit objects in JSP?

Q.11 What is a Cookie? For what purpose they are used? Write a Perl script to write a cookie to a client's machine.

Q.12 What is Java Bean ? Explain the structure of Java Bean.

Q.13 Give the syntax of doPost and doGet methods. What are the key components of a JSP?

****



XML

IN THIS CHAPTER, YOU WILL LEARN

- Introduction to XML
- Relationship between HTML, SGML and XML
- Structure of XML Document
- Well Formed and Valid XML Document
- Embedding XML into HTML documents
- Converting XML to HTML for Display
- Displaying XML using CSS and XSL, rewriting HTML as XML
- The future of XML.

10.1 INTRODUCTION TO XML

XML is a Extensible markup language for documents containing well structured information. Structured information contains any type of content (words, pictures, etc.) and some indication of what role that content plays (for example, content in a section heading has a different significance from content in a footnote, which means something different than content in a figure caption or content in a database table, etc.). Almost all documents have some structure.

A markup language is a mechanism to identify the document structures. The XML is used to define a standard way to add markup to documents. ***It was designed to carry data, not to display data. Its tags are not predefined. You must define your own tags.*** XML is designed to be self-descriptive. XML is a formal recommendation from the World Wide Web Consortium (W3C) similar to the language of today's Web pages, the Hypertext Markup Language (HTML).

10.2 Relationship between HTML, SGML, and XML

First you should know that SGML (Standard Generalized Markup Language) is the basis for both HTML and XML. SGML is an international standard (ISO 8879) that was published in 1986.

Second, you need to know that XHTML is XML. "XHTML 1.0 is a reformulation of HTML 4.01 in XML, and combines the strength of HTML 4 with the power of XML."

Thirdly, XML is NOT a language, it is rules to create an XML based language. Thus, XHTML 1.0 uses the tags of HTML 4.01 but follows the rules of XML.

The Document: A typical document is made up of three layers:

- Structure
- Content
- Style

- **Structure :** would be the documents title, author, paragraphs, topics, chapters, head, body etc.
- **Content :** Content is the actual information that composes a title, author, paragraphs etc.
- **Style :** Style is how the content within the structural elements are displayed such as font color, type and size, text alignment etc.

Markup : HTML, SGML, and XML all markup content using tags. The difference is that SGML and XML mainly deal with the relationship between content and structure, the structural tags that markup the content are not predefined (you can make up your own language), and style is kept TOTALLY separate; HTML on the other hand, is a mix of content marked up with both structural and stylistic tags. HTML tags are predefined by the HTML language.

By mixing structure, content and style you limit yourself to one form of presentation and in HTML's case that would be in a limited group of browsers for the World Wide Web. By separating structure and content from style, you can take one file and present it in multiple forms. XML can be transformed to HTML/XHTML and displayed on the Web, or the information can be transformed and published to paper, and the data can be read by any XML aware browser or application.

SGML (Standard Generalized Markup Language) : Historically, Electronic publishing applications such as Microsoft Word, Adobe PageMaker or QuarkXpress, "marked up" documents in a proprietary format that was only recognized by that particular application. The document markup for both structure and style was mixed in with the content and was published to only one media, the printed page.

These programs and their proprietary markup had no capability to define the appearance of the information for any other media besides paper, and really did not describe very well the actual content of the document beyond paragraphs, headings and titles. The file format could not be read or exchanged with other programs, it was useful only within the application that created it.

Because SGML is a nonproprietary international standard it allows you to create documents that are independent of any specific hardware or software. The document structure (what elements are used and their relationship to each other) is described in a file called the DTD (Document Type Definition). The DTD

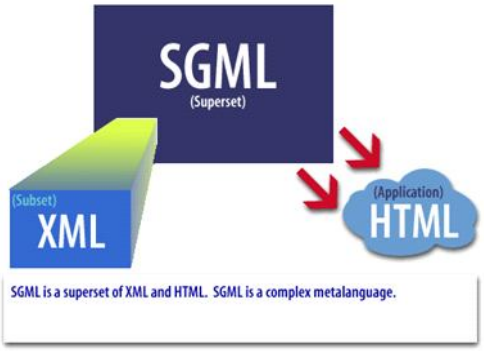


Fig. 10.1

defines the relationships between a document's elements creating a consistent, logical structure for each document. SGML is good for handling large-scale, long-term information management needs and has been around for more than a decade as the language of defense contractors and the electronic publishing industry. Because SGML is very large, powerful, and complex it is hard to learn and understand and is not well suited for the Web environment.

XML (Extensible Markup Language) : XML is a "restricted form of SGML" which removes some of the complexity of SGML. XML like SGML, retains the flexibility of describing customized markup languages with a user-defined document structure (DTD) in a non-proprietary file format for both storage and exchange of text and data both on and off the Web.

As mentioned before, XML separates structure and content from style and the structural markup tags can actually describe the content because they can be customized for each XML based markup language. A good example of this is the Math Markup Language (MathML) which is an XML application for describing mathematical notation and capturing both its structure and content.

Until MathML, the ability to communicate mathematical expressions on the Web was limited to mainly displaying images (JPG or GIF) of the scientific notation or posting the document as a PDF file. MathML allows the information to be displayed on the Web, and makes it available for searching, indexing, or reuse in other applications.

HTML (Hypertext markup Language) : HTML is a single, predefined markup language that forces Web designers to use it's limiting and lax syntax and structure. The HTML standard was not designed with other platforms in mind, such as Web TV's, mobile phones or PDAs. The structural markup does little to describe the content beyond paragraph, list, title and heading.

XML breaks the restricting chains of HTML by allowing people to create their own markup languages for exchanging information. The tags can be descriptive of the content and authors decide how the document will be displayed using style sheets (CSS and XSL). Because of XML's consistent syntax and structure, documents can be transformed and published to multiple forms of media and content can be exchanged between other XML applications. HTML was useful in the part it has played in the success of the Web but has been outgrown as the Web requires more robust, flexible languages to support it's expanding forms of communication and data exchange.

Features of XML –

1. **XML is Just Plain Text :** XML is nothing special. It is just plain text. Software that can handle plain text can also handle XML. However, XML-aware applications can handle the XML tags specially. The functional meaning of the tags depends on the nature of the application. With XML You Invent Your Own Tags. The tags are "invented" by the author of the XML document. That is because the XML language has no predefined tags.

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

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

2. **XML is Not a Replacement for HTML :** XML is a complement to HTML. It is important to understand that XML is not a replacement for HTML. In most web applications, XML is used to transport data, while HTML is used to format and display the data. My best description of XML is this: XML is a software and hardware independent tool for carrying information.

3. **XML is a W3C Recommendation :** XML became a W3C Recommendation 10. February 1998.

4. **XML is Everywhere :** We have been participating in XML development since its creation. It has been amazing to see how quickly the XML standard has developed, and how quickly a large number of software vendors have adopted the standard. XML is now as important for the Web as HTML was to the foundation of the Web. XML is everywhere. It is the most common tool for data transmissions between all sorts of applications, and is becoming more and more popular in the area of storing and describing information.

5. **XML Separates Data from HTML :** If you need to display dynamic data in your HTML document, it will take a lot of work to edit the HTML each time the data changes. With XML, data can be stored in separate XML files. This way you can concentrate on using HTML for layout and display, and be sure that changes in the underlying data will not require any changes to the HTML. With a few lines of JavaScript, you can read an external XML file and update the data content of your HTML. You will learn more about this in a later chapter of this tutorial.

6. **XML Simplifies Data Sharing :**In the real world, computer systems and databases contain data in incompatible formats. XML data is stored in plain text format. This provides a software- and hardware-independent way of storing data. This makes it much easier to create data that different applications can share.

7. **XML Simplifies Data Transport :** With XML, data can easily be exchanged between incompatible systems. One of the most time-consuming challenges for developers is to exchange data between incompatible systems over the Internet.

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

8. **XML Simplifies Platform Changes :**

   Upgrading to new systems (hardware or software platforms), is always very

time consuming. Large amounts of data must be converted and incompatible data is often lost. XML data is stored in text format. This makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.

9. XML Makes Your Data More Available :

Since XML is independent of hardware, software and application, XML can make your data more available and useful. Different applications can access your data, not only in HTML pages, but also from XML data sources. With XML, your data can be available to all kinds of "reading machines" (Handheld computers, voice machines, news feeds, etc), and make it more available for blind people, or people with other disabilities.

10. XML is Used to Create New Internet Languages :

- A lot of new Internet languages are created with XML.
- Here are some examples:
- XHTML the latest version of HTML
- WSDL for describing available web services
- WAP and WML as markup languages for handheld devices

10.3 Structure of XML Document

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE document system "tutorials.dtd">
<!-- Here is a comment -->
<?xml-stylesheet type="text/css" href="myStyles.css"?>
<tutorials>
<tutorial>
<name>XML Tutorial</name>
<url>http://www.quackit.com/xml/tutorial</url>
</tutorial>
<tutorial>
<name>HTML Tutorial</name>
<url>http://www.quackit.com/html/tutorial</url>
</tutorial>
</tutorials>
```

The following table provides an explanation of each part of the XML document in the above example:

Prolog (optional)	**XML Declaration**	<?xml version="1.0" encoding="UTF-8" standalone="no"?>
	Document Type Definition (DTD)	<!doctype document system "tutorials.dtd">
	Comment	<!-- Here is a comment -->
	Processing Instructions	<?xml-stylesheet type="text/css" href="myStyles.css"?>
	White Space	
Elements & Content (required)	**Root element opening tag**	<tutorials>
	Child elements and content	<tutorial> <name>XML Tutorial</name> <url>http://www.quackit.com/xml/tutorial</url> </tutorial> <tutorial> <name>HTML Tutorial</name> <url>http://www.quackit.com/html/tutorial</url> </tutorial>
	Root element closing tag	</tutorials>

Prolog (optional) : Right at the top of the document, we have a prolog (also spelt prologue). A prolog is optional, but if it is included, it should become at the beginning of the document. The prolog can contain things such as the XML declaration, comments, processing instructions, white space, and document type declarations. Although the prolog (and everything in it) is optional, it's recommended that you include the XML declaration in your XML documents.

1. XML Declaration : The XML declaration indicates that the document is written in XML and specifies which version of XML. The XML declaration, if included, must be on the first line of the document. The XML declaration can also specify the language encoding for the document (optional) and if the application refers to external entities (optional). In our example, we specify that the document uses UTF-8 encoding (although we don't really need to as UTF-8 is the default), and we specify that the document refers to external entities by using standalone="no". This is not a standalone document as it relies on an external resource (i.e. the DTD).

2. Document Type Definition (DTD) : The DTD defines the rules of your XML document. Although XML itself has rules, the rules defined in a DTD are specific to your own needs. More specifically, the DTD allows you to specify the names of the elements that are allowed in the document, which elements are allowed to be nested inside other elements, and which elements can only contain data.

The DTD is used when you validate your XML document. Any application that uses the document must stop processing if the document doesn't adhere to the DTD.

DTDs can be internal (i.e. specified within the document) or external (i.e. specified in an external file). In our example, the DTD is external.

3. Comments : XML comments begin with <!-- and end with -->. Similar to HTML comments, XML comments allow you to write stuff within your document without it being parsed by the processor. You normally write comments as an explanatory note to yourself or another programmer. Comments can appear anywhere within your document.

4. Processing Instructions : Processing instructions begin with <? and end with?>. Processing instructions are instructions for the XML processor. Processing instructions are not built into the XML recommendation. Rather, they are processor-dependant so not all processors understand all processing instructions.

5. White Space : White space is simply blank space created by carriage returns, line feeds, tabs, and/or spaces. White space doesn't affect the processing of the document, so you can choose to include whitespace or not.

Speaking of white space, there is a special attribute (xml:whitespace) that you can use to preserve whitespace within your elements (but we won't concern ourselves with that just now).

10.4 Elements & Content (Required)

This is where the document's content goes. It consists of one or more elements, nested within a single root element.

- **Root Element Opening Tag :** All XML documents must have one (and only one) root element. All other elements must be nested inside this root element. In other words, the root element must contain all other elements within the document. Therefore, the first tag in the document will always be the opening tag of the root element (the closing tag will always be at the bottom of the document).

- **Child Elements and Content :** These are the elements that are contained within the root element. Elements are usually represented by an opening and closing tag. Data and other elements reside between the opening and closing tag of an element. Although most elements contain an opening and closing tag, XML allows you to use empty elements. An empty element is one without a closing tag. You might be familiar with some empty elements used in HTML such as the <img> element or the
 element. In XML, you must close empty elements with a forward slash before the > symbol. For example,
.

  Elements can also contain one or more attributes. An attribute is a name/ value pair, that you place within an opening tag, which allows you to provide extra information about an element. You may be familiar with attributes in HTML. For example, the HTML img tag requires the src attribute which specifies the location of an image (eg,).

- **Root Element Closing Tag :** The last tag of the document will always be the closing tag of the root element. This is because all other elements are nested inside the root element.

10.4.1 XML Elements

XML elements are represented by tags. Elements usually consist of an opening tag

and a closing tag, but they can consist of just one tag. Opening tags consist of <, followed by the element name, and ending with >. Closing tags are the same but have a forward slash inserted between the less than symbol and the element name.

Example:

```
<tag>Data</tag>
```

Example of empty tag:

```
<tag />
```

The following syntax rules are important to note, especially if you're used to working with HTML where you don't usually need to worry about these rules.

1. **All Elements Must Be Closed Properly :** If you're familiar with HTML, you will know that some HTML tags don't need to be closed. In XML however, you must close all tags. This is usually done in the form of a closing tag where you repeat the opening tag, but place a forward slash before the element name (i.e. </child>). If you are using an empty element (i.e. one with no closing tag), you need to place a forward slash before the greater than symbol at the end of the tag (i.e. <child />).

Example for opening/closing tags:

```
<child>Data</child>
```

Example for empty elements:

```
<child attribute="value" />
```

2. **Tags Are Case Sensitive :** All tags must be written using the correct case. XML sees <tutorial> as a different tag to <Tutorial>

Wrong:

```
<Tutorial>XML</tutorial>
```

Right:

```
<Tutorial>XML</Tutorial>
<tutorial>XML</tutorial>
<TUTORIAL>XML</TUTORIAL>
```

3. **Elements Must Be Nested Properly :** You can place elements inside other elements but you need to ensure each element's closing tag doesn't overlap with any other tags.

Wrong:

```
<tutorial>
<name>XML</tutorial>
</name>
```

Right:

```
<tutorial>
<name>XML</name>
</tutorial>
```

XML Attributes : The previous lesson covered the syntax rules related to XML elements. XML elements can also contain attributes. You use attributes within your elements to provide more information about the element. These are represented as name/value pairs.

Example:

```
<tag attribute="value">Data</tag>
```

It's important to remember the following syntax rules when using attributes.

1. **Quotes :** You must place quotation marks around the attribute's value.

Wrong:

```
<tutorials type=Web>
<tutorial>
```

Right:

```
<tutorials type="Web">
<tutorial>
```

```
<name>XML</name>                    <name>XML</name>
</tutorial>                         </tutorial>
</tutorials>                        </tutorials>
```

2. Shorthand Is Prohibited : Attributes must contain a value. Some HTML coders like to use shorthand, where if you provide the attribute name without a value, it will equal true. This is not allowed in XML.

Wrong:	***Right:***
`<tutorials published>`	`<tutorials published="true">`
`<tutorial>`	`<tutorial>`
`<name>XML</name>`	`<name>XML</name>`
`</tutorial>`	`</tutorial>`
`</tutorials>`	`</tutorials>`

10.5 DTD (DOCUMENT TYPE DEFINITION)

A DTD is a set of rules that defines what tags appear in a XML document, what attributes the tags may have and what a relationship the tags have with each other. When an XML document is processed, it is compared within the DTD to be sure it is structured correctly and all tags are used in the proper manner. This comparison process is called validation and it is performed by a tool called parser. DTD can be used to define a document type for SGML (Standard Generalized Markup Language) documents. Since XML (Extensible Markup Language) is a subset of SGML, DTD can also be used to define a document type for XML documents. If an XML document is said to be valid against a DTD document type, all elements, attributes and entities in the XML document must meet their declared formats described in the DTD document type.

Main features of DTD

1. DTD is a simple language with only 4 types of statements: DOCTYPE, ELEMENT, ATTLIST, and ENTITY.
2. One DOCTYPE statement defines one document type.
3. Within the DOCTYPE statement, one or more ELEMENT statements, some ATTLIST statements and some ENTITY statements are included to define details of the document type.
4. DTD statements that define the document type can be included inside the XML file.
5. DTD statements that define the document type can be stored as a separate file and linked to the XML file.
6. Validation of XML files against their document types can be done by XML validation tools.

i) Internal DTDs : Internal DTD (markup declaration) are inserted within the doctype declaration. DTDs inserted this way are used in the that specific document. This might be the approach to take for the use of a small number of tags in a single

document, as in this example:

```
<?xml version="1.0"?>
<!DOCTYPE film [
     <!ENTITY COM "Comedy">
     <!ENTITY SF "Science Fiction">
     <!ELEMENT film (title+,genre,year)>
     <!ELEMENT title (#PCDATA)>
     <!ATTLIST title
         xml:lang NMTOKEN "EN"
         id ID #IMPLIED>
     <!ELEMENT genre (#PCDATA)>
     <!ELEMENT year (#PCDATA)>
     ]>
<film>
     <title id="1">Tootsie</title>
     <genre>&COM;</genre>
     <year>1982</year>
     <title id="2">Jurassic Park</title>
     <genre>&SF;</genre>
</film>
```

ii) External DTD : DTDs can be very complex and creating a DTD requires a certain amount of work. DTDs are stored as ASCII text files with the extension '.dtd'. In the following example we assume, that the previously internal DTD was saved as a separate file (under the namefilm.dtd), and is therefore now referred to as external definition (external DTD):

```
<?xml version="1.0"?>
<!DOCTYPE film SYSTEM "film.dtd">
<film>
<title id="1">Tootsie</title>
<genre>&COM;</genre>
<year>1982</year>
<title id="2">Jurassic Park</title>
<genre>&SF;</genre>
<year>1993</year>
</film>
```

10.6 Well formed XML documents

An XML document needs to be well formed. Well formed means that the document applies to the syntax rules for XML.

The rules: To be well formed a document needs to comply with the following rules:

- It contains a root element.
- All other elements are children of the root element
- All elements are correctly paired
- The element name in a start-tag and an end-tag are exactly the same
- Attribute names are used only once within the same element

10.7 Valid XML documents

To be of practical use, an XML document needs to be valid. To be valid an XML document needs to apply to the following rules:

- The document must be well formed.
- The document must apply to the rules as defined in a Document Type Definition (DTD), (More on DTD's in the next page).

If a document is valid, it's clearly defined what the data in the document really means.

There's no possibility to use a tag that's not defined in the DTD. Companies that exchange XML-documents can check them with the same DTD. Because a valid XML document is also well formed, there's no possibility for typo's in the tags.

A valid XML-document has a structure that's valid. That's the part you can check. There's no check for the content.

Difference between Valid XML and Well-Formed Xml : A valid document conforms to semantic rules but can be also user defined, while a simple well formed xml structure only respects basic xml syntax rules.

10.8 Embedding XML into HTML Document

One serious proposal is for HTML documents to support the inclusion and processing of XML data. This would allow an author to embed within a standard HTML document some well delimited, well defined XML object. The HTML document would then be able to support some functions based on the special XML markup. This strategy of permitting "islands" of XML data inside an HTML document would serve at least two purposes:

1. To enrich the content delivered to the web and support further enhancements to the XML-based content models.
2. To enable content developers to rely on the proven and known capabilities of HTML while they experiment with XML in their environments. The result would look like this:

```
<HTML>
<body>
<!-- some typical HTML document with
<h1>, <h2>, <p>, etc. -->
<xml>
```

```
<!-- The <xml> tag introduces some XML-compliant markup for some specific
purpose. The markup is then explicitly terminated with the </xml> tag.
The user agent would invoke an XML processor only
on the data contained in the <xml></xml> pair. Otherwise the user agent
would process the containing document as an HTML document. -->
</xml>
<!-- more typical HTML document markup -->
</body>
</html>
```

10.9 Displaying XML Document using CSS

CSS stands for Cascading Style Sheets. Styles define how to display HTML elements. Styles are normally stored in Style Sheets. Styles were added to HTML 4.0 to solve a problem. External Style Sheets can save a lot of work. External Style Sheets are stored in CSS files. Multiple style definitions will cascade into one.

A Cascading Style Sheet is a file that contains instructions for formatting the elements in an XML document.

Creating and linking a CSS to your XML document is one way to tell browser how to display each of document's elements. An XML document with an attached CSS can be open directly in Internet Explorers. You don't need to use an HTML page to access and display the data.

There are two basic steps for using a css to display an XML document:

- Create the CSS file.
- Link the CSS sheet to XML document.

Creating CSS File : CSS is a plain text file with .css extension that contains a set of rules telling the web browser how to format and display the elements in a specific XML document. You can create a css file using your favorite text editors like Notepad, Wordpad or other text or HTML editor as show below:

Linking : To link to a style sheet you use an XML processing directive to associate the style sheet with the current document. This statement should occur before the root node of the document.

```
<?xml-stylesheet type="text/css" href="styles/general.css">
```

The two attributes of the tag are as follows:

1. **href:** The URL for the style sheet.
2. **type:** The MIME type of the document begin linked, which in this case is text/css.

MIME stands for Multipart Internet Mail Extension. It is a standard which defines how to make systems aware of the type of content being included in e-mail messages.

general.css	The css file is designed to attached to the XML document as below:

employees
{
background-color:
#ffffff;
width: 100%;
}
id
{
display: block;
margin-bottom: 30pt;
margin-left: 0;
}
name
{
color: #FF0000;
font-size: 20pt;
}
city,state,zipcode
{
color: #0000FF;
font-size: 20pt;
}
	<?xml version="1.0" encoding="utf-8" standalone="no"?>
	<!--This xml file represent the details of an employee-->
	<?xml-stylesheet type="text/css"
	href="styles/general.css">
	<employees>
	<employee id="1">
	<name>
	<firstName>Mohit</firstName>
	<lastName>Jain</lastName>
	</name>
	<city>Karnal</city>
	<state>Haryana</state>
	<zipcode>98122</zipcode>
	</employee>
	<employee id="2">
	<name>
	<firstName>Rahul</firstName>
	<lastName>Kapoor</lastName>
	</name>
	<city>Ambala</city>
	<state>Haryana</state>
	<zipcode>98112</zipcode>
	</employee>

10.10 Displaying XML Document using XSL

It is a language for expressing stylesheets. It consists of two parts:

- A language for transforming XML documents (XSLT)
- An XML vocabulary for specifying formatting semantics

An XSL stylesheet specifies the presentation of a class of XML documents by describing how an instance of the class is transformed into an XML document that uses the formatting vocabulary. Like CSS an XSL is linked to an XML document and tell browser how to display each of document's elements. An XML document with an attached XSL can be open directly in Internet Explorers. You don't need to use an HTML page to access and display the data. There are two basic steps for using a css to display an XML document:

- Create the XSL file.
- Link the XSL sheet to XML document.

Creating XSL file : XSL is a plain text file with .css extension that contains a set of rules telling the web browser how to format and display the elements in a specific XML document. You can create a css file using your favorite text editors like Notepad, Wordpad or other text or HTML editor as show below:

Linking : To link to a style sheet you use an XML processing directive to associate the style sheet with the current document. This statement should occur before the root node of the document.

```
<?xml-stylesheet type="text/xsl" href="styles/general.xsl">
```

The two attributes of the tag are as follows:

1. **href:** The URL for the style sheet.
2. **type:** The MIME type of the document begin linked, which in this case is text/css.

MIME stands for Multipart Internet Mail Extension. It is a standard which defines how to make systems aware of the type of content being included in e-mail messages.

general.xsl The css file is designed to attached to the XML document as below:

general.xsl	The css file is designed to attached to the XML document as below:

employees
{
background-color:
#ffffff;
width: 100%;
}
id
{
display: block; margin-
bottom: 30pt; margin-
left: 0;
}
name
{
color: #FF0000;
font-size: 20pt;
}
city,state,zipcode
{
color: #0000FF;
font-size: 20pt;
}
	<?xml version="1.0" encoding="utf-8" standalone="no"?>
	<!--This xml file represent the details of an employee-->
	<?xml-stylesheet type="text/xsl"
	href="styles/general.xsl">
	<employees>
	<employee id="1">
	<name>
	<firstName>Mohit</firstName>
	<lastName>Jain</lastName>
	</name>
	<city>Karnal</city>
	<state>Haryana</state>
	<zipcode>98122</zipcode>
	</employee>
	<employee id="2">
	<name>
	<firstName>Rahul</firstName>
	<lastName>Kapoor</lastName>
	</name>
	<city>Ambala</city>
	<state>Haryana</state>
	<zipcode>98112</zipcode>
	</employee>
	</employees>

10.11 Converting XML to HTML for Display

There exist several ways to convert XML to HTML for display on the Web.

1. **Using HTML alone :** If your XML file is of a simple tabular form only two levels deep then you can display XML files using HTML alone.
2. **Using HTML + CSS :** This is a substantially more powerful way to transform XML to HTML than HTML alone, but lacks the full power and flexibility of the methods listed below.
3. **Using HTML with JavaScript :** Fully general XML files of any type and complexity can be processed and displayed using a combination of HTML and JavaScript. The advantages of this approach are that any possible transformation and display can be carried out because JavaScript is a fully general purpose programming language. The disadvantages are that it often requires large,

complex, and very detailed programs using recursive functions (functions that call themselves repeatedly) which are very difficult for most people to grasp.

4. **Using XSL and Xpath :** XSL (eXtensible Stylesheet Language) is considered the best way to convert XML to HTML. The advantages are that the language is very compact, very sophisticated HTML can be displayed with relatively small programs, it is easy to re-purpose XML to serve a variety of purposes, it is non-procedural in that you generally specify only what you wish to accomplish as opposed to detailed instructions as to how to achieve it, and it greatly reduces or eliminates the need for recursive functions. The disadvantages are that it requires a very different mindset to use, and the language is still evolving so that many XSL processors in the Web servers are out of date and newer ones must sometimes be invoked through DOS.

10.12 The Future of XML

The future of XML is still unclear because of conflicting views of XML users. Some say that the future is bright and holds promise. While others say that it is time to take a break from the continuous increase in the volume of specifications. In the past five years, there have been substantial accomplishments in XML. XML has made it possible to manage large quantities of information which don't fit in relational database tables, and to share labeled structured information without sharing a common Application Program Interface (API). XML has also simplified information exchange across language barriers.

But as a result of these accomplishments, XML is no longer simple. It now consists of a growing collection of complex connected and disconnected specifications. As a result, usability has suffered.

This is because it takes longer to develop XML tools. These users are now rooting for something simpler. They argue that even though specifications have increased, there is no clear improvement in quality. They think in might be better to let things be, or even to look for alternate approaches beyond XML. This will make XML easier to use in the future. Otherwise it will cause instability with further increase in specifications. The other side paints a completely different picture. They are ready for further progress in XML. There have been discussions for a new version, XML 2.0. This version has been proposed to contain the following characteristics:

- Elimination of DTDS
- Integration of namespace
- XML Base and XML Information Set into the base standard

Research is also being carried out into the properties and use cases for binary encoding of the XML information set.

Future of XML Applications : The future of XML application lies with the Web and Web Publishing. Web applications are no longer traditional. Browsers are now integrating games, word processors and more. XML is based in Web Publishing, so the future of XML is seen to grow as well.

SHORT QUESTION ANSWERS

Q.1 What is a markup language?

Ans. A markup language is a set of words and symbols for describing the identity of pieces of a document (for example 'this is a paragraph', 'this is a heading', 'this is a list', 'this is the caption of this figure', etc). Programs can use this with a style sheet to create output for screen, print, audio, video, Braille, etc.

Q. 2 What is XML?

Ans. XML is the Extensible Markup Language. It improves the functionality of the Web by letting you identify your information in a more accurate, flexible, and adaptable way. It is extensible because it is not a fixed format like HTML (which is a single, predefined markup language). Instead, XML is actually a meta language-a language for describing other languages-which lets you design your own markup languages for limitless different types of documents. XML can do this because it's written in SGML, the international standard meta language for text document markup (ISO 8879).

Q. 3 Aren't XML, SGML, and HTML all the same thing?

Ans. Not quite; SGML is the mother tongue, and has been used for describing thousands of different document types in many fields of human activity, from transcriptions of ancient Irish manuscripts to the technical documentation for stealth bombers, and from patients' clinical records to musical notation. SGML is very large and complex, however, and probably overkill for most common office desktop applications.

XML is an abbreviated version of SGML, to make it easier to use over the Web, easier for you to define your own document types, and easier for programmers to write programs to handle them. It omits all the complex and less-used options of SGML in return for the benefits of being easier to write applications for, easier to understand, and more suited to delivery and interoperability over the Web. But it is still SGML, and XML files may still be processed in the same way as any other SGML file (see the question on XML software).

HTML is just one of many SGML or XML applications-the one most frequently used on the Web. Technical readers may find it more useful to think of XML as being SGML- rather than HTML++.

Q.4 What are the benefits of XML?

Ans. There are many benefits of using XML on the Web :

1. Simplicity- Information coded in XML is easy to read and understand, plus it can be processed easily by computers.

2. Openness- XML is a W3C standard, endorsed by software industry market leaders.
3. Extensibility - There is no fixed set of tags. New tags can be created as they are needed.
4. Self-description- In traditional databases, data records require schemas set up by the database administrator. XML documents can be stored without such definitions, because they contain meta data in the form of tags and attributes.
5. Contains machine-readable context information- Tags, attributes and element structure provide context information that can be used to interpret the meaning of content, opening up new possibilities for highly efficient search engines, intelligent data mining, agents, etc.
6. Facilitates the comparison and aggregation of data - The tree structure of XML documents allows documents to be compared and aggregated efficiently element by element.
7. Can embed multiple data types - XML documents can contain any possible data type - from multimedia data (image, sound, video) to active components (Java applets, ActiveX).

Q.5 What is the difference between XML and HTML?

HTML	XML
HTML is for displaying purpose.	whereas XML is for data representation.
HTML is used to mark up text so it can be displayed to users.	XML is used to mark up data so it can be processed by computers.
HTML describes both structure (e.g. <p>, <h2>,) and appearance (e.g. , <font>, <i>)	XML describes only content, or “meaning”
HTML uses a fixed, unchangeable set of tags	In XML, you make up your own tags

XML is no way clashes with HTML, since they are for two different purposes.

Q.6 Do I have to know HTML or SGML before I learn XML?

Ans. No, although it's useful because a lot of XML terminology and practice derives from two decades' experience of SGML. Be aware that 'knowing HTML' is not the same as 'understanding SGML'. Although HTML was written as an SGML application, browsers ignore most of it (which is why so many useful things don't work), so just because something is done a certain way in HTML browsers does not mean it's correct, least of all in XML.

Q.7 What is the structure of XML document ?

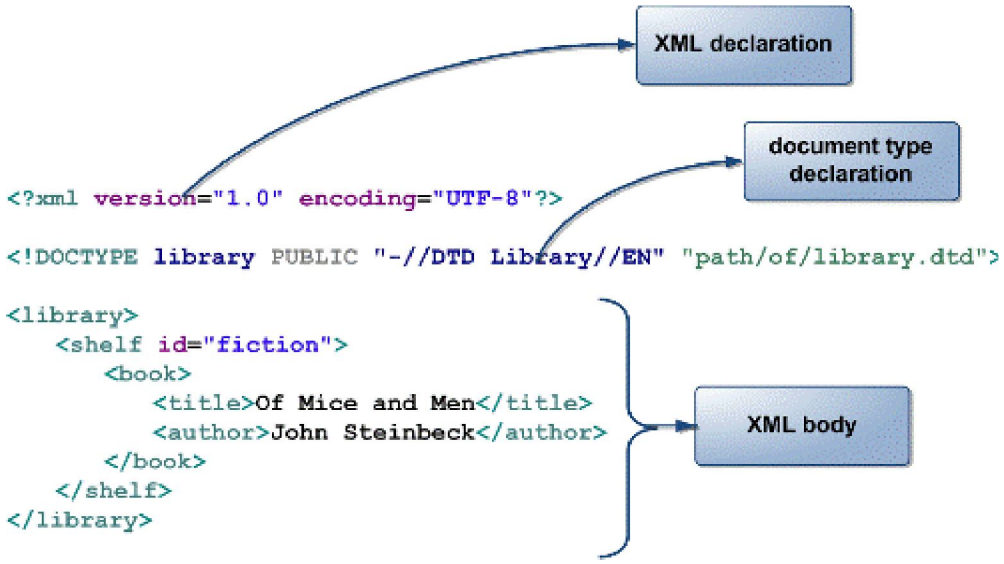


Fig. 10.2

Q.8 Define XML Elements, Attributes with example?

Ans. There are no rules about when to use attributes or when to use elements.

For Example:

1. <person sex="female">
   <firstname>Anna</firstname>
   <lastname>Smith</lastname>
   </person>
2. <person>
   <sex>female</sex>
   <firstname>Anna</firstname>
   <lastname>Smith</lastname>
   </person>

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

Q.9 What is a well-formed XML document?

Ans. If a document is syntactically correct it can be called as well-formed XML documents. A well-formed document conforms to XML's basic rules of syntax:

1. Every open tag must be closed.
2. The open tag must exactly match the closing tag: XML is case-sensitive.
3. All elements must be embedded within a single root element.

4. Child tags must be closed before parent tags.
5. A well-formed document has correct XML tag syntax, but the elements might be invalid for the specified document type.

Q.10 What is a valid XML document?

Ans. If a document is structurally correct then it can be called as valid XML documents. A valid document conforms to the predefined rules of a specific type of document:

1. These rules can be written by the author of the XML document or by someone else.
2. The rules determine the type of data that each part of a document can contain.

Note:*Valid XML document is implicitly well-formed, but well-formed may not be valid*

Q.11 How does the XML structure is defined?

Ans. XML document will have a structure which has to be defined before we can create the documents and work with them. The structural rules can be defined using many available technologies, but the following are popular way of doing so-

1. Document Type Definition (DTD) 2. Schema

Q.12 What is DTD?

Ans. A Document Type Definition (DTD) defines the legal building blocks of an XML document. It defines rules for a specific type of document, including:

1. Names of elements, and how and where they can be used
2. The order of elements
3. Proper nesting and containment of elements
4. Element attributes

To apply a DTD to an XML document, you can:

1. Include the DTD's element definitions within the XML document itself.
2. Provide the DTD as a separate file, whose name you reference in the XML document.

Q.13 What is XML Schema?

Ans. An XML **Schema** describes the structure of an XML instance document by defining what each element must or may contain.XML Schema is expressed in the form of a separate XML file.

- XML Schema provides much more control on element and attribute datatypes.
- Some datatypes are predefined and new ones can be created.

```
<xsd:schema  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element  name="test">
<xsd:complexType>
```

Q. 14 What are differences between DTDs and Schema?

Schema	DTD
Schema document is an XML document i.e., the structure of an XML document is specified by another XML document.	DTDs follow SGML syntax.
Schema supports variety of dataTypes similar to programming language.	In DTD everything is treated as text.
In Schema, It is possible to inherit and create relationship among elements.	This is not possible in DTD without invalidating existing documents.
In Schema, It is possible to group elements and attributes so that they can be treated as single logical unit.	Grouping of elements and attributes is not possible in DTD.
In Schemas, it is possible to specify an upper limit for the number of occurrences of an element	It is not possible to specify an upper limit of an element in DTDs

Q.15 Define XML namespace?

Ans. XML Namespaces provide a method to avoid element name conflicts. XML namespace is a collection of element type and attribute names. A reasonable argument can be made that XML namespaces don't actually exist as physical or conceptual entities.

Q.16 What is XSL?

Ans. XSL is a language for expressing style sheets. An XSL style sheet is a file that describes the way to display an XML document. Using XSL stylesheets, we can separate the XML document content and its styling. An XSL style sheet begins with the XML declaration:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet> defines that the document is an XSLT style sheet document.
```

The <xsl:template> element defines a template.

Q.17 Define CSS and XSL.

Ans. XSL is a language for expressing style sheets. An XSL style sheet is a file that describes the way to display an XML document.

Cascading Style Sheets is an answer to the limitations of HTML, where the structure of documents was defined and not the display. CSS formats documents for display in browsers that support it.

Q.18 What Is the Relation between XHTML and XML?

Ans. XML (Extensible Markup Language) is a generic markup language to organize generic information into a structured document with embedded tags.

XHTML is entirely based on XML. You can actually say that XHTML is a child language of XML.

MULTIPLE CHOICE QUESTIONS

1. SGML stands for –

a. Standard Generalized Markup Language
b. Structured General Markup Language
c. Standard Graphics Mapping Language
d. Standard General Markup Link

2. HTML and XML are markup languages

a. Specially development for the web
b. Are based on SGML
c. Are versions of SGML
d. Independent of SGML

3. XML stands for

a. Extra Markup Language
b. Excellent Markup Links
c. Extended Markup Language
d. Extended Marking Links

4. XML uses

a. user define tags
b. pre-defined tags
c. both predefined and user-defined tags
d. Extended tags used in HTML and makes them powerful

5. In order to interpret XML documents one should

a. Use standardized tags
b. Have a document type definition which defines the tags
c. Define the tags separately
d. Specify tag filename

6. The advantages of XML over HTML are

(i) It allows processing of data stored in web-pages
(ii) It uses meaningful tags which aids in understanding the nature of a document
(iii) Is simpler than HTML
(iv) It separates presentation and structure of document

a. (i),(ii) and (iii)
b. (i),(ii) and(iv)
c. (ii),(iii) and (iv)
d. (i),(iii) and (iv)

7. XSL definition is used along with XML definition to specify

a. The data types of the contents of XML document
b. The presentation of XML document
c. The links with other documents
d. The structure of XML document

8. XLL definition is used along with XML to specify

a. The data types of the contents of XML document
b. The presentation of XML document
c. The links with other documents
d. The structure of XML document

9. DTD definition is used along with XML to specify

a. The data types of the contents of XML document

b. The presentation of XML document
c. The links with other documents
d. The structure of XML document

10. Output of XML document can be viewed in a

a. Word Processor b. Web browser
c. Notepad d. None of the above

11. What is the correct way of describing XML data?

a. XML uses a DTD to describe data
b. XML uses a description node to describe data
c. XML uses XSL to describe the data
d. XML uses a validator to describe the data

12. Comments in XML document is given by:

a. <?_ _ _ _> b. <!_ _ _ _!>
c. <!_ _ _ _> d. </_ _ _ _>

13. Which statement is true?

a. An XML document can have one root element
b. An XML document can have one child element
c. XML elements have to be in lower case
d. All of the above

KEY TO OBJECTIVE QUESTIONS

1.a	2.b	3.c	4.a	5.b	6.b	7.b
8.c	9.a	10.b	11.c	12.c	13.a	

IMPORTANT QUESTIONS

Q.1 How HTML, SGML, and XML are linked to each other ?
Q. 2 Explain the following: a) HTML vs XML b) Converting XML to HTML for display.
Q.3 Explain with example how to embed XML into HTML documents?
Q.4 What are the different ways to use XML ?
Q.5 What are the different ways to display XML documents ?
Q.6 Explain the following terms : a) Embedding XML into HTML documents.
b) Converting XML to HTML for display.
Q.7 What is Well Formed Document ? How it is made valid ?
Q.8 What are differences between DTDs and Schema?
Q.9 Explain the structure of XML documents.
Q.10 What is XML Elements and Attributes ?
Q.11 a) What is XSL? What are the steps to transform XML into HTML using XSL?
b) How is XSL different from CSS? Discuss.
Q.12 Why is XML such an important development?
Q.13 What are the XML rules for distinguishing between the content of a document and the XML markup element?

****

HTML 4.01 / XHTML 1.0 Tags Reference

Tag	Description
<!--...-->	Specifies a comment
<!DOCTYPE>	Specifies the document type
<a>	Specifies an anchor
<abbr>	Specifies an abbreviation
<acronym>	Specifies an acronym
<address>	Specifies an address element
<applet>	Deprecated. Specifies an applet
<area>	Specifies an area inside an image map
<b>	Specifies bold text
<base>	Specifies a base URL for all the links in a page
<basefont>	Deprecated. Specifies a base font
<bdo>	Specifies the direction of text display
<bgsound>	Specifies the background music
<big>	Specifies big text
<blink>	Specifies a text which blinks
<blockquote>	Specifies a long quotation
<body>	Specifies the body element
 	Inserts a single line break
<button>	Specifies a push button
<caption>	Specifies a table caption
	Deprecated. Specifies centered text
<cite>	Specifies a citation
<code>	Specifies computer code text
<col>	Specifies attributes for table columns
<colgroup>	Specifies groups of table columns
<comment>	Puts a comment in the document
<dd>	Specifies a definition description
<del>	Specifies deleted text
<dfn>	Specifies a definition term
<dir>	Deprecated. Specifies a directory list
	Specifies a section in a document
<dl>	Specifies a definition list
<dt>	Specifies a definition term

	Specifies emphasized text
<embed>	Deprecated. Embeds an application in a document
<fieldset>	Specifies a fieldset
<font>	Deprecated. Specifies text font, size, and color
<form>	Specifies a form
<frame>	Specifies a sub window (a frame)
<frameset>	Specifies a set of frames
<h1> to <h6>	Specifies header 1 to header 6
<head>	Specifies information about the document
<hr>	Specifies a horizontal rule
<html>	Specifies an html document
<I>	Specifies italic text
<iframe>	Specifies an inline sub window (frame)
<ilayer>	Specifies an inline layer
<img>	Specifies an image
<input>	Specifies an input field
<ins>	Specifies inserted text
<isindex>	Deprecated. Specifies a single-line input field
<kbd>	Specifies keyboard text
<keygen>	Generate key information in a form
<label>	Specifies a label for a form control
<layer>	Specifies a layer
<legend>	Specifies a title in a fieldset
	Specifies a list item
<link>	Specifies a resource reference
<map>	Specifies an image map
<marquee>	Create a scrolling-text marquee
<menu>	Deprecated. Specifies a menu list
<meta>	Specifies meta information
<multicol>	Specifies a multicolumn text flow
<nobr>	No breaks allowed in the enclosed text
<noembed>	Specifies content to be presented by browsers that do not support the <embed> tag
<noframes>	Specifies a noframe section
<noscript>	Specifies a noscript section
<object>	Specifies an embedded object
	Specifies an ordered list

<optgroup>	Specifies an option group
<option>	Specifies an option in a drop-down list
<p>	Specifies a paragraph
<param>	Specifies a parameter for an object
<plaintext>	Deprecated. Render the raminder of the document as preformatted plain text
<pre>	Specifies preformatted text
<q>	Specifies a short quotation
<s>	Deprecated. Specifies strikethrough text
<samp>	Specifies sample computer code
<script>	Specifies a script
<select>	Specifies a selectable list
<spacer>	Specifies a white space
<small>	Specifies small text
	Specifies a section in a document
<strike>	Deprecated. Specifies strikethrough text
	Specifies strong text
<style>	Specifies a style definition
<sub>	Specifies subscripted text
<sup>	Specifies superscripted text
<table>	Specifies a table
<tbody>	Specifies a table body
<td>	Specifies a table cell
<textarea>	Specifies a text area
<tfoot>	Specifies a table footer
<th>	Specifies a table header
<thead>	Specifies a table header
<title>	Specifies the document title
<tr>	Specifies a table row
<tt>	Specifies teletype text
<u>	Deprecated. Specifies underlined text
	Specifies an unordered list
<var>	Specifies a variable
<wbr>	Indicate a potential word break point within a <nobr> section
<xmp>	Deprecated. Specifies preformatted text.

****

INTERNET GLOSSARY

Acceptable Use Policy: A former set of formal rules that govern how a network, application or piece of information may be used. See also netiquette, Terms of Service.

Active X: Software technology developed by Microsoft for including applications into HTML pages. Lack of security let many people prefer Java over Active X.

Address: An address on the Internet is described as a uniform resource locator, which can be used for any type of addressing, such as e-mails (mailto:info@gallery-net.com), web pages (http://www.news.com/) and ftp sites (ftp://ftp.netscape.com/pub/communicator). Instead of using domain names, it is also possible to use IP addresses. See also ftp, e-mail, IP, uniform resource locator, web page.

Address Resolution Protocol: Used primarily with IP-Network Layer to resolve addresses.

ADSL: See a symmetric digital subscriber line. Advanced Research Projects Agency Network: A computer network that has been developed in the late 60s by the US Department of Defense to allow communication in a post-nuclear war age. Predecessor of the Internet. See also internet.

Ad-Server: A program or server that is responsible for handling the banner advertisements for several web sites. These servers offer statistics about visits and movements of customers. They offer also functionality, such as banner rotation, so that a single customer will not see a certain banner twice, when visiting the same web page.

Ad Transfer: An ad transfer is the successful arrival of a customer at the site of the banner advertisement. See also banner advertisement.

Agent: Application that acts for a customer by completing transactions, seeking information or prices or communicating with other agents and customers.

AI: See artificial intelligence.

Anonymous FTP: See ftp.

Anti-aliasing: Process used to remove jagged edges in computerized graphics.

American Standard Code For Information Interchange: A standard for the representation of upper and lower-case Latin letters, numbers and punctuation on computers. There are 128 standard ASCII codes which are represented by a 7 digit binary code. The other 128 codes are used differently on most computers. In order to display non-Latin codes, Unicode is used in most cases. See also binary code, unicode.

API: See application program interface.

Applet: Java programs that are embedded into HTML pages. Applets are restricted, in such a way that they are, for example, not allowed to read and write to the hard disk of the user without explicit permission. See also HTML, java, servlet.

Application: A program, which is self-contained and that executes a set of well-defined tasks under user control.

Application Program Interface: Interface, which allows the communication between programs, networks and databases.

Archie: Piece of software for finding files on anonymous FTP sites. It searches only for file names and has been replaced by more powerful web-based search engines. See also anonymous FTP.

ARP: See address resolution protocol.

ARPANet: See advanced research projects agency network.

Artificial Intelligence: A branch of computer science that studies how to endow computers with capabilities of human intelligence.

ASCII: See american standard code for information interchange.

Asymmetric Digital Subscriber Line: ADSL is becoming the alternative to an ISDN line. It allows much higher bandwidths over a standard digital telephone line. It needs to be configured similar to a leased line in such a way that it can connect only to ISPs that are near you. A typical ADSL setup allows download speeds up to 1.5 megabits per second (about 200 kilobits per seconds), but upload is restricted to 128 kilobits per second (similar to two ISDN lines). ADSL works asynchronously, therefore the different up- and download speeds. See also bit, bps, ISDN, leased line.

Asynchronous Transfer Method: A fast, intelligent hardware switch which can support voice, data, image, and video. Cell-switching (as opposed to packet) technology which replaces variable-length packets now in use with uniform (53 byte) cells. It promises any-to-any connectivity and networks that scale easily from a few nodes to global deployment. Combines packet switching's efficient use of bandwidth with circuit switching's minimal delays.

ATM: See asynchronous transfer method.

Attached File: A file, for example an application, image or sound, that is embedded into an email message. See also e-mail.

Audio Video Interleaved: Windows format for saving video with sound.

Authentication: The process of verifying a person.

Authorization: The process of allowing access to a system to a person.

AUP: See acceptable use policy.

Avatar: Three-dimensional representation or digital actor of a customer in a web shop or in a chat room.

AVI: See audio video interleaved.

Backbone: The top level of a hierarchical network. Major pathway within a network offering the highest possible speed and connecting all major nodes. The main pipes along which data is transferred. See also network, nodes.

Bandwidth: The maximum size of information that can be sent through a connection at a given time. Usually measured in bits per second (bps). See also bps, bit, T-1.

BASIC: See beginners all-purpose symbolic instructional code.

Baud: It is commonly used in the same way as bits per second. See also bit, bps, modem.

Beginners All-purpose Symbolic Instructional Code: A computer language that is easy-to-learn and highly flexible, which was invented at Dartmouth University.

Beta: A pre-release of an application that is made available for the purposes of testing.

Binary: Mathematical base 2, or numbers composed of a series of zeros and ones. Since zero's and one's can be easily represented by two voltage levels on an electronic device, the binary number system is widely used in digital computing.

Binary Digit: A single digit number in base 2 (therefore 0 or 1). The smallest unit for computerized data. See also byte, kilobyte, megabyte.

Bit: See binary digit.

Bits per second: Measurement unit for transferring data. A 33.6 modem can move 33.600 bits per second. See also bandwidth, bit.

Bookmark: A file that contains references to web pages that you have already visited, which then can be organized and used to return to a particular page later on.

Boot: To start up or reset a computer. When a computer is booted the operating syis loaded. There are two different types of booting a computer. A cold boot means that the computer needs to powered up from an off state and a warm boot means that all data in the memory is erased and the operating system is loaded from start. See also memory, operating system.

Bps: See bits per second.

Browser: Client application that is able to display various kinds of Internet resources. See also Client, homepage, internet explorer, mosaic, netscape, URL, WWW.

Bug: A programming error that causes a malfunction of the computer software or hardware. See also hardware, software.

Byte: Eight bits for a byte which is used to represent a single ASCII character, for example. See also bit.

Central Processing Unit: The main chip inside every computer that is used to run the operating system and the application software.

Certificate Authority: Issuer of digital certificates, used for encrypting communication and signing documents. See also digital certificates, digital signatures.

CGI: See common gateway interface.

Chat: Direct communication over the Internet with multiple persons. Other than e-mail, responses are made in real-time. See also IRC.

Checksum: A special calculation applied to a piece of information. If the information

is transmitted and the calculation achieves the same result, then the transmission was successful.

Client: Application that resides on the customers computer and contacts a server to communicate. Examples: IRC clients, Web Clients. See also IRC, web.

Clipboard: A piece of memory that stores information only temporarily. See also memory.

CODEC: Program or device that COmpresses/DECompresses digital video.

Common Gateway Interface: A standard that describes how a web browser passes on information to a web server. CGI programs are able to read the information, process it and pass the results back to the web browser.

Compiler: A program that translates a programming language into machine code. See also programming language, machine code.

Compression: Technology to reduce the size of files and save bandwidth. See also bandwidth, lossy compression, non-lossy compression.

Connection: Established path for exchanging information.

Cookie: Piece of information that is stored in the browser and can be retrieved by the server that placed the information there. This piece of information can be used to identify a user, for example.

CPU: See central processing unit.

Cracker: A person who tries to break the copy protection of software.

Cyberspace: First used in Neuromancer by William Gibson. It is used to describe the Internet. See also internet.

Data Encryption Key: A string used to mathematically encode a message so that it can only be decrypted by someone with either the same key (symmetric encryption) or with a related key (asymmetric encryption).

Data Encryption Standard: Encryption scheme, developed by IBM in the 1970s.

Database: Collection of data formatted in a special way, to make it easier to retrieve a particular piece of information.

DHCP: See dynamic host configuration protocol.

DHTML: See dynamic hypertext markup language.

Dial-Up: A temporary connection between two computers established over a phone line.

Digital Certificate: File containing information about its owner that can be used to identify the owner. See also certificate authority, SSL.

Digital Signal Processor: A separate processor, built into some sound cards, that relieves audio processing from the computer's CPU.

Disk Operating System: Outdated operating system with a command line interface. See also operating system.

Dithering: If a color is not available, it can be made from the available colors by placing a pattern of colors next to each other to visually mix. For instance the illusion of “orange” can be made by placing red and yellow pixels next to each other.

DNS: See domain name system.

Domain Name: The name of a computer connected to the Internet. The Domain name is used to form a URL. See also URL.

Domain Name System: Database that links IP addresses and domain names. See also domain name, IP.

DOS: See disk operating system.

Download : To receive files from another computer on the Internet by actively requesting it.

DSP: See digital signal processor.

DVD: See digital versatile disk.

Dynamic Host Configuration Protocol: Internet standard, based on RFC 1541, for the automatic allocation of IP addresses.

Dynamic Hypertext Markup Language: An extension to HTML, which allows a better user interaction and introduces dynamic web page creation.

E-Mail: See electronic mail.

Electronic Data Interchange: A standard for the inter-organizational computer-to-computer exchange of structured information.

Electronic Mail: Exchange of digital documents via the Internet.

Encryption: Procedure to render a message illegible to anyone who is not authorized to read it.

Ethernet: Standard for connecting computers on an Intranet. See also bandwidth, intranet.

Extranet: Extranets are extended Intranets to share information with business partners over the Internet in a very secure way. See also intranets.

FAQ: See frequently asked questions.

File Transfer Protocol: Internet protocol to move files from one Internet site to another one. Public FTP servers allow the up- and download of files, creating public file archives.

Firewall: A tool to seperate an Intranet from the Internet by disallowing connections on certain ports, making the Intranet very secure. See also network, intranet.

Font: Typographic style such as Times Roman or Helvetica.

Forward: Sending on an e-mail to a third person. See also e-mail.

Frame: HTML tag that allows the browser window to be segmented into several sections. See also browser, HTML.

Freeware: Software that is available to anybody without the need for paying a fee, while the author retains the copyright.

Frequently Asked Questions: A web page that lists and answers the most common question on a particular subject.

FTP: See file transfer protocol.

Fuzzy Search: Finds matches even if the keyword is misspelled or only partially spelled.

Gateway: Architecture for bridging between two networks that work with different protocols.

GIF: See graphic interchange format.

Gigabyte: 1024 Megabytes, but some use 1000 Megabytes, as it is easier to calculate with. See also Byte, Megabyte.

Graphic Interchange Format: Image format, very common on the Internet. See also JPEG, PNG.

Gopher: Internet protocol for presenting menus of downloadable documents or files. It is still around, but has no real importance anymore. See also HTTP, Hypertext, WWW.

Graphical User Interface: Graphical environment to simplify the use of the operating system and applications.

Grep: Unix command to scan files for patterns, also used as a synonym for fast manual searching.

GUI: See graphical user interface.

Hacker: Persons who spend their time breaking into systems and networks in order to steal, change or delete data that does not belong to them.

Homepage: The main page on a web server.

Host: See server.

HTML: See hypertext markup language.

HTTP: See hypertext transport protocol.

Hypertext: Web documents that contain links to other documents.

HyperText Markup Language: The language for developing documents for the World Wide Web. See also client, server, WWW.

HyperText Transport Protocol: The protocol for transporting files from a web server to a web browser. See also client, server, WWW.

Icon: Mnemonic convention to replace functional names by images.

IDE: See integrated drive electronics.

Index: A searchable database of documents created automatically or manually by a search engine.

Integrated Services Digital Network: Digital version of the good old analogue telephone line.

International Organization for Standardization: A federation of national standards bodies such as BSI and ANSI.

Internet: The computer network for business and leisure based on the TCP/IP protocol. All other computer networks have become irrelevant. Evolved from ARPAnet. See also ARPANet, network, TCP/IP.

Internet Protocol: See TCP/IP.

Internet Service Provider: Company providing access to the Internet.

Internet Society: Non-governmental international organization for global co-operation and co-ordination of the Internet and its technologies and applications.

Intranet: Private network that is based on the same technologies as the Internet, but restricted to a certain user group. See also internet, network.

IP: See internet protocol.

ISDN: See integrated services digital network.

ISO: See international organization for standardization.

ISOC: See internet society.

ISP: See internet service provider.

JAR: See Java Archive.

Java: Programming language developed by Sun with cross-platform neutrality, object-orientation and networking in mind. See also applet, JDK.

Java Archive: A file format used to bundle all components required by a Java applet. JAR files simplify the downloading of applets since all the components (.class files, images, sounds, etc.) can be packaged into a single file.

Java Development Kit: Basic development package from Sun distributed for free in order to write, test and debug Java programs. See also applet, java.

JavaScript: Scripting language developed by Netscape that allows interaction within HTML pages. See also HTML, scripting.

JDK: See Java development kit.

JIT: See just in time.

Joint Photographic Experts Group: Multi-company commission that develops new image formats.

JPEG: Image format for the Internet using lossy compression algorithms. See also joint photographic experts group.

Just In Time: The concept of reducing inventories by working closely with suppliers to co-ordinate delivery of materials just before their use in the manufacturing or supply process.

Kilobyte: 1024 Bytes, sometimes 1000 Bytes. See also bit, byte.

LAN: See also local area network.

Leased-Line: A permanently established phone line that is used to offer twenty-four hour access to the Internet.

Local Area Network: Computer network limited to a certain location. See also ethernet, WAN.

Login: Account name to gain access to a system. See also password.

Megabyte: 1024 Kilobytes, sometimes 1000 Kilobytes. See also kilobyte, byte, bit.

MIME: See multipurpose internet mail extensions.

Modem: See modulator demodulator.

Modulator Demodulator: Device between computer and phone line that converts computer signals to a form that can be used to transport the data over telephone networks.

Multipurpose Internet Mail Extensions: Format for attaching binary files to e-mails. Enables multi-part/multimedia messages to be sent over the Internet. This standard was developed by the Internet Engineering Task Force (IETF). See als binhex, uuencode.

Network: The connection of two or more computers in order to share resources is a network.

Newsgroup: Discussion group on USENET. See also USENET.

Node: A device connected to a network.

Object Oriented Programming: Art of programming independent pieces of code, which are then able to interact with each other.

Offline: Not connected to the Internet.

Online: Connected to the Internet.

OOP: See object oriented programming.

Operating System: Software that is loaded right after the boot time. It provides the basic functionality to run applications, based on a single set of instructions.

Packet: The smallest unit for transmitting data over the Internet. Data is broken up into packets, sent over the network and then reassembled at the other end.

Password: Secret code to identify a user when logging onto a system.

Perl: Powerful scripting language, often used to write CGI scripts.

Plug-ins: Software that adds functionality to commercial applications, such as the Netscape browser or Adobe's Photoshop.

Point of Presence: Local access to the services of an ISP. See also ISP.

POP: See point of presence.

POP3: See post office protocol.

Port: Interface for accessing services on a server.

Portal: Point of entry web site to the Internet.

Post Office Protocol: Protocol for receiving mails via a client.

Protocol: Rules how computers and applications interact.

Proxy Server: A proxy server retrieves documents on demand from a server and passes them on to a client. The advantage with a proxy server is that it normally caches documents. It is considerably faster to retrieve documents from the proxy rather than directly from a web server, especially if someone else has already retrieved that particular document.

Query: Request for information from a database.

Queue: Sequence of objects.

RAM: See random access memory.

Random Access Memory: Memory, that is used for executing applications and storing documents while working on them.

Remote Login: Logging into a computer system from remote.

Root: The administrator account that has super-user rights on a system. See also sysop.

Router: A device to handle the connection between two or more networks. See also network.

Search Engine: Web service that allows you to query a database for keywords and returns matching web pages.

Secure Sockets Layer: Protocol invented by Netscape to encrypt communication between web browser and server. It provides privacy, authentication and integrity.

Server: device that provides one or more services to several clients over a network. See also client, network.

Servlet: A Java application that runs on a server. The term usually refers to a Java applet that runs within a Web server environment. This is analogous to a Java applet that runs within a Web browser environment.

Simple Mail Transport Protocol: The protocol to send electronic mail over the Internet. See also e-mail.

Smart Card: Plastic card of credit card size with an embedded microchip. The chip can contain digital money and personal information about the owner.

SMTP: See simple mail transport protocol.

Spam: Inappropriate use of e-mail and postings by sending information and advertising to people who did not request them.

Spider: See web crawler.

SQL: See structured query language.

Structured Query Language: The preferred programming language for communication with databases.

SSL: See secure sockets layer.

TCP / IP: See transmission control protocol/internet protocol.

Telnet: Program to perform a remote login to another computer.

Terabyte: 1024 or 1000 Gigabytes. See also byte, kilobyte.

Thin Client: A cut-down network terminal with no local processing power.

Transmission Control Protocol / Internet Protocol: A set of protocols that are the foundation of the Internet, which enable the communication between computers.

Triple-dub: Net-language for "WWW".

Trojan Horse: A program that seems to be harmless and starts harmful functions after it has been installed.

Unicode: Text encoding scheme including international characters and alphabets.

Uniform Resource Locator: Addressing scheme on the Internet to locate Internet resources.

Unix: Operating System, developed in the early seventies.

URL: See uniform resource locator.

USENET: A decentralized world-wide system for newsgroups. See also newsgroups.

Virtual Memory System: A multiuser, multitasking, virtual memory operating system for the VAX series from Digital Equipment.

Virus: Malicious piece of code that can be hidden in programs and destroy data on a computer.

WAN: See wide area network.

Web: See world wide web.

Web Crawler: Service that scans web documents and adds them to a database. After having indexed one page it follows all links and indexes them as well. See also search engine.

Webmaster: The person in charge of a web server. Most web servers will allow mails to be sent to the webmaster. The web master of http://www.foobar.org/ can be reached at webmaster@foobar.org, for example. See also postmaster.

Wide Area Network: A network that is distributed over several locations. See also LAN.

World Wide Web: The part of the Internet, which is accessible through a web browser. The Web is not the Internet, but a subset.

****