Chapter 1

Web Basics and Overview

The Web is an Internet-based distributed information system. Anyone with a computer connected to the Internet can easily retrieve information by giving a Web address or by simply clicking a mouse button. The Web is a great way to disseminate information and making it available 24/7. Information can also be collected from Web users and customers through online forms. Maintainers and administrators can control and update Web content from anywhere on the Web. All these make the Web a powerful tool for mass communication, e-business and e-commerce.

Compared with TV, radio, news papers, and magazines, putting the word out on the Web is relatively simple and inexpensive. But a website is much more than such one-way communication media. It can be a virtual office or store that is always open and supported by workers from anywhere.

Web service companies offer free Web space and tools to generate simple personal or even business Web pages. But, well-designed and professionally implemented websites are much more involved. Even then, expertly produced websites are still much more cost-effective than other means of mass communication. For business and commerce, the cost of a website is negligible when compared to building and operating a brick-and-mortar office or store. Once in-place, a website is a *store that never closes* and that is very attractive. People take great pains in building an office or store to project the right image and to serve the needs of customers. Likewise, well-informed businesses will insist on professionally architected, designed and implemented websites. Nothing less will do.

As a communication medium, the Web consists of these major components

- Networks—The local-area and wide-area networks connecting computers world-wide forming the Internet.
- Clients—Web browsers that enable end-users to access the Web.
- Servers—Constantly running programs that serve up information to the Web.
- Documents—Web pages, mostly coded in HTML, that supply information on the Web.
- Protocols—The *Hyper Text Transfer Protocol* HTTP that Web clients and servers use to talk to one another and the TCP/IP (*Transmission Control Protocol*) on which HTTP depends.

A basic understanding of these components and how they work together lays a good foundation for *Web Design and Programming*. Let's begin by taking a look at networking.

1.1 About Networking

A *computer network* is a high-speed communications medium connecting many, possibly dissimilar, computers or *hosts*. A network is a combination of computer and telecommunication hardware and software. The purpose is to provide fast and reliable information exchange among the hosts and between *processes*, or executing programs, on different hosts. The Web is one of the most widely used Internet services. Others include: e-mail, file transfer, audio/video streaming, and login to remote hosts, just to name a few. The Web also provides convenient ways to tap into these other Internet services.

A network extends greatly the powers of the connected hosts. Modern computers and networks are so integrated it is hard to tell where the computer ends and where the network begins. The view “The Network is the Computer.” is more valid than ever before and companies slow to adopt this view are scrambling to implement *network-centric* solutions to their corporate needs. A website can work wonders for companies, organizations, governments, and even individuals.

Networking Protocols

In order for programs and computers from different vendors, under different operating systems, to communicate on a network, a detailed set of rules and conventions must be established for all parties to follow. Such rules are known as *networking protocols.* Protocols govern such details as

- address format of hosts and processes
- data format
- manner of data transmission
- sequencing and addressing of messages
- initiating and terminating connections
- establishing remote services
- accessing remote services
- network security

Thus, in order for a process on one host to communicate with another process on a different host, both processes must follow the same protocol. A protocol is usually viewed as having logical layers that come between the process and the networking hardware. The corresponding layers on different hosts perform complementary tasks to make the connection between the communicating processes.

Among common networking protocols, the *Internet Protocol* (IP) suite[1] is the most widely used. IP is the basic protocol for the *Internet* (Section 1.2) which is, by far, the most predominant worldwide network. The Web is a service that uses HTTP (the Hyper Text Transfer Protocol), which is based on Internet protocols.

Networking protocols are no mystery. Think about the *protocol for making telephone calls*. You (a client process) must pick up the phone, listen to the dial tone, dial a valid telephone number, wait for the other side (the server process) to pick up the phone. Then you must say *hello* and identify yourself, etc. This is a protocol from which you can't deviate if you want the call to be made successfully through the telephone network. And it is clear why such a protocol is needed. Same goes for a computer program getting to talk to another through a computer network. The design of efficient and effective networking protocols for different network services is an important area in computer science.

If your computer system is on a network, chances are that you are already connected to the Internet. This means you have the ability to, almost instantaneously, reach across great distances to obtain information, exchange messages, retrieve data, interact with others, do literature searches, and much more all without leaving the seat in front of your workstation. If your computer is not directly connected to a network but has a telephone or cable modem, then you can reach the Internet through *Internet Access Providers* (IAP).

1.2 The Internet

Internet is a global network that connects IP networks. The linking of computer networks is called *internetworking*, hence the name Internet. Internet links all kinds of organizations around the world–universities, government offices, corporations, libraries, supercomputer centers, research labs, and even individual homes. The number of connections on the Internet is large and growing rapidly.

The Internet evolved from the ARPANET[2], a US Defense Advanced Research Projects

[1]Including TCP, UDP, and others.

[2]The ARPANET was started in the late 1960s as an experimental facility for reliable military networking.

Agency (DARPA) sponsored network that developed the IP as well as the higher-level TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) networking protocols. The architecture and protocol were designed to support a reliable and flexible network that can endure war-time attacks.

The transition of ARPANET to Internet took place in the late 1980s as NSFNET, the US National Science Foundation's network of universities and supercomputing centers, helped create an explosive number of IP-based local and regional networks and connections. The NSFNET remains an important component of Internet. The Internet is so dominant now that it has virtually eliminated all historical rivals such as Bitnet and Decnet.

The *Internet Corporation for Assigned Names and Numbers* (ICANN, `www.icann.org`) is a non-profit organization responsible for IP address space allocation, protocol parameter assignment, domain name system management, and root server system management functions.

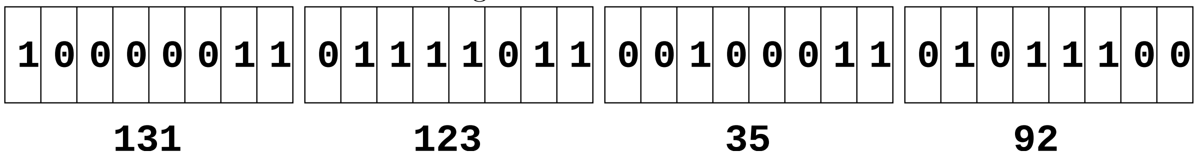
Network Addresses

An address to a host computer is like a phone number to a telephone. Every host on the Internet has a unique network address that identifies the host for communication purposes. The addressing scheme is an important part of a network and its protocol. For the Internet, each host has a unique *IP address* represented by 4 bytes in a 32-bit quantity. For example, `monkey`, a host at Kent State, has the IP address `131.123.35.92` (Figure 1.1). This *dot notation* (or *quad notation*) gives the decimal value (0 to 255) of each byte[3]. The IP address is similar to a telephone number in another way: the leading digits are like area codes and the trailing digits are like local numbers.

Because of their numerical nature, the dot notation is easy on machines but hard on users. Therefore, each host also has a unique *domain-based name* composed of words, rather like a postal address. For example, the domain name for `monkey` is `monkey.cs.kent.edu`

[3]To accommodate the explosive growth of the Internet, the next generation IP (IPv6) will support 128-bit addresses.

Figure 1.1: IP Address



(at Dept. of Computer Science, Kent State University). With the domain names, the entire Internet host name space is recursively divided into disjoint domains. The address for `monkey` puts it in the `kent` subdomain within `edu`, the *top-level domain* (TLD) for educational institutions. Other TLDs include `org` (non-profit organizations), `gov` (government offices), `mil` (military installations), `com` (commercial outfits), `net` (network service providers), `uk`, (United Kindom), `cn` (China), etc. Within a local domain (e.g. `cs.kent.edu`) you can refer to machines by their host name alone (e.g. `monkey`, `dragon`, `tiger`), but the full address must be used for machines outside. Further information on Internet domain names can be found in Section 1.10.

The ICANN accredits *domain name registrars* who register domain names for clients so they stay unique. All network applications accept a host address given either as a domain name or an IP address. In fact, a domain name is first translated to a numerical IP address before being used.

Packet Switching

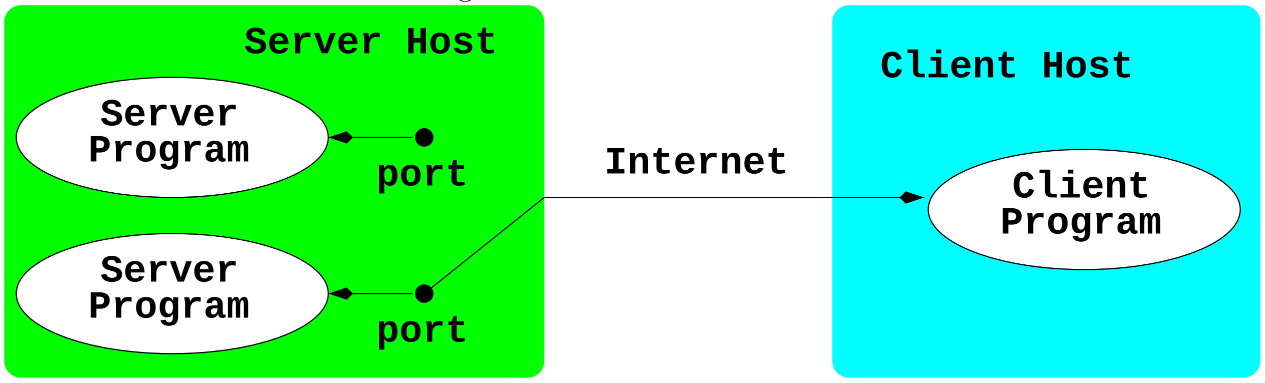
Data on the Internet are sent and received in *packets*. A packet envelops transmitted data with address information so the data can be *routed* through intermediate computers on the network. Because there are multiple routes from the source to the destination host, the Internet is very reliable and can operate even if parts of the network are down.

Client and Server

Most commonly, a network application involves a *server* and a *client* (Figure 1.2):

- A *server* process provides a specific service on a host machine that offers such a service.

Figure 1.2: Client and Server



Example services are remote host access (`TELNET`), file transfer (`FTP`), and the World-Wide Web (`HTTP`). Each *Internet standard service* has its own unique *port number* that are identical across all hosts. The port number together with the Internet address of a host identifies a particular server (Figure 1.2) anywhere on the Network. For example, `FTP` has port number 21, `Telnet` 23, and `HTTP` 80.

- A *client* process on a host connects with a server on another host to obtain its service. Thus, a client program is the agent through which a particular network service can be obtained. Different agents are usually required for different services.

A Web browser such as Netscape is an HTTP client. It runs on your computer to access Web servers on any Internet hosts.

1.3 The Domain Name System

As stated in Section 1.2, every host on the Internet has a unique *IP address* and a domain name. The *network name space* is the set of all host names and changes dynamically with time due to addition/deletion of hosts, regrouping of local work groups, reconfiguration of subparts of the network, maintenance of systems and networks, and so on. So, new domain names, new IP addresses, and new domain-to-IP associations can be introduced in the name space at any time without central control. The *domain name system* (DNS) provides a distributed database service that supports dynamic update and retrieval of information contained in the name space (Figure 1.3). A network client program (e.g. the Netscape Navigator browser) will normally use the DNS to obtain address information for a target host before making contact with a server. The dynamic DNS also supplies a general mechanism for retrieving many kinds of information about hosts and even individual users.

Here are points to note about the DNS name space:

- The DNS organizes the entire Internet name space into a big tree structure. Each node of the tree has a *label* and a list of *resources*.

- Labels are character strings (currently not case-sensitive) and sibling labels must be distinct. The root is labeled by the empty string. Immediately below the root are the top-level domains: `edu`, `com`, `gov`, `net`, `org`, and so on. TLDs also include country names–`at` (Austria), `ca` (Canada), `cn` (China), for example. Under `edu`, for example, there are subdomains `berkeley`, `kent`, `mit`, `uiuc`, and so on (Figure 1.4).

- A full *domain name* of a node is a dot-separated list of labels leading from the node up to the root (`cs.kent.edu.`, for example).

- A *relative domain name* is a prefix of a full domain name indicating a node relative to a domain of origin. Thus, the familiar `cs.kent.edu` is actually a name relative to the root.

- A label is the formal or *canonical* name of a domain. Alternative names, called *aliases*, are also allowed. For example, the main Web server host `info` has the alias `www` so it is also known as `www.cs.kent.edu`. To move the Web server to a different host, a local system manager simply reassigns the alias to another host.

Figure 1.3: Domain to IP

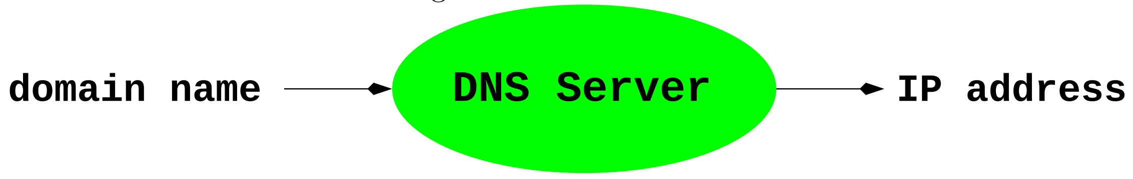
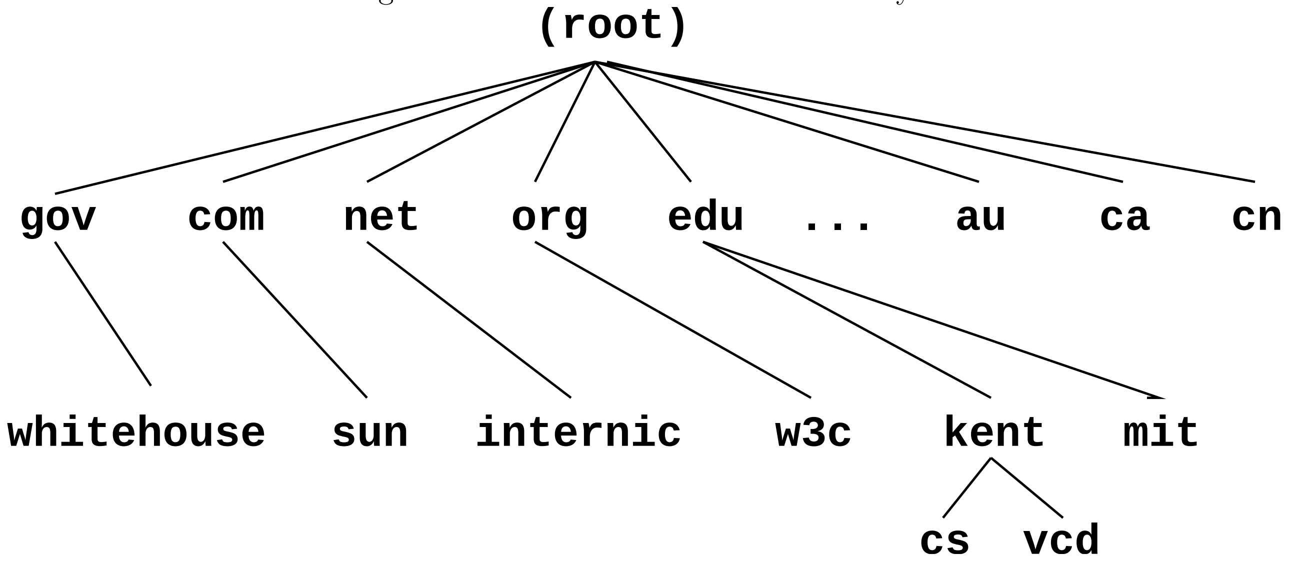


Figure 1.4: The Domain Name Hierarchy



See Section 1.10 for more information on the DNS and name servers.

1.4 The Web

There is no central control or administration for the Web. Anyone can potentially put material on the Web and retrieve information from it. The Web consists of a vast collection of *documents* that are located on computers throughout the world. These documents are created by academic, professional, governmental, and commercial organizations as well as by individuals. The documents are prepared in special formats and retrieved through *server programs* on each computer that provides Web service. Each Web document can contain (potentially many) links to other documents served by different servers in other locations and therefore become part of a *web* that spans the entire globe. New materials are being put on the Web continuously and instant access to this collection of information can be enormously advantageous. As the Web grows explosively, MIT of the USA and INRIA (the French National Institute for Research in Computer Science and Control) have agreed to become joint hosts of the *W3 Consortium* which is supported by industry and will further develop Web related standards, protocols, and services.

A *Web browser* is a program that helps users obtain information from the Web. Given the location of a target document, a browser connects to the correct Web server, retrieves and displays the desired document. You can click on *links* in a document to obtain other documents. Using a browser you can retrieve information provided by *Web servers* anywhere on the Internet.

Many different Web browsers are available. *Mosaic*, developed at the US National Center for Supercomputing Applications (NCSA), is the original browser with a convenient graphical user interface. Today, widely used Web browsers are Netscape's *Netscape Navigator* (NN) and Microsoft's *Internet Explorer* (IE). RealOne is an audio/video media player and Web browser from RealNetworks. Other browsers include IBM's *WebExplorer*, JavaSoft's *HotJava*, W3C's *Amaya*, *Mozilla*, *Opera*, just to name a few. Web browsers compete to offer speed and convenience for the user and are evolving with time.

Typically a browser supports the display of HTML files and images in standard formats. Helper applications or plug-ins can augment a browser to treat pages with multimedia contents such as audio, video, animation, and mathematical formulas.

Hypertext

A Web browser communicates with a Web server through an efficient *Hypertext Transfer Protocol* (HTTP) designed to work with *hypertext* and *hypermedia* documents that may contain regular text, images, audio, and video. Native Web pages are written in the *Hypertext Markup Language* (HTML) and saved usually in files with the `.html` (or `.htm`) name suffix.

HTML organizes Web page contents (text, graphics, and other media data) and allows *hyperlinks* to other pages anywhere on the Web. Clicking on such a link causes your Web browser to follow it and retrieve another page. The Web employs an open addressing scheme allowing links to objects and services provided by Web, email, file transfer, audio/video, and newsgroup servers. Thus, the Web space is a superset of many popular Internet services. Consequently, a Web browser provides the ability to access a wide variety of information and services on the Internet.

URL

The Web uses *Uniform Resource Locators* (URLs) to identify (locate) resources (files and services) available on the Internet. A URL may identify a host, a server port, and the target file stored on that host. URLs are used, for example, by browsers to retrieve information and by HTML to link to other resources.

A full URL usually has the form

scheme`://`*server*`:`*port*`/`*pathname*

The *scheme* part indicates the information service type and therefore the protocol to use. Common schemes include the following:

- `http`—Service is Web. The file located is retrieved by the Web-defined hypertext transfer protocol (HTTP).
- `ftp`—Service is FTP. The URL locates a file, a directory, or an FTP server. For example,

  `ftp://ftp1.mcom.com/netscape/`

  The protocol is the *file transfer protocol*.
- `file`—Service is the local file system. The URL locates a file on the same host.
- `mailto`—Service is email. The URL is simplified and identifies an email address to send email via the Internet.
- `telnet`—Service is TELNET. The URL names a target host for remote login.
- `news`—The URL locates a USENET newsgroup.

Many other schemes can be found at `www.w3.org/addressing/schemes`.

The *server* identifies a host and a server program. The optional port number is needed only if the server does not use the default port (e.g. 21 for `FTP` and 80 for `HTTP`). The remainder of the URL, when given, is a *file pathname.* If this pathname has a trailing `/` character, it represents a directory, rather than a data file. The suffix (`.html`, `.txt`, `.jpg`, etc.) of a data file indicates the file type. The pathname can also lead to an executable program that dynamically produces an HTML or other valid file to return.

Within an HTML document you can link to another document served by the same Web server by giving only the *pathname* part of the URL. Such URLs are *partially specified.* A partial URL with a `/` prefix (e.g. `/file_xyz.html`) refers to a file under the *server root*, the top-level directory controlled by the Web server. A partial URL without a leading `/` points to a file relative to the location of the document that contains the URL in question. Thus, a simple `file_abc.html` refers to that file in the same directory as the current document. When building a website, it is advisable to use URL relative to the current page as much as possible.

Accessing Information on the Web

You can directly access any Web document, directory, or service by giving its URL in the `Location` box of a browser. When given a URL that specifies a directory, a Web server usually returns an *index file* (typically `index.html`) for that directory, Otherwise, it may return a list of the filenames in that directory.

The Web contains a vast amount of useful information in a loosely organized fashion. However, locating sites with something related to what you are looking for may not be simple. Fortunately, there are *search engines* that collect information available on the Web and establish easy-to-search databases. These search engines continuously update their databases and can be enormously helpful in locating information. Newly established websites usually submit their URLs to popular search engines so the new sites will be included in the search databases.

Yahoo! (`www.yahoo.com`) is among the first such engines. Here are some others:

```
www.google.com     www.lycos.com
www.excite.com     www.askjeeves.com
```

Table 1.1: Content Types and File Suffixes

Content Type	File Suffix
`text/html`	`html htm`
`image/jpeg`	`jpeg jpg jpe`
`audio/basic`	`au snd`
`audio/mpeg`	`mpeg mp2 mp3`
`audio/x-realaudio`	`ra`
`audio/x-wav`	`wav`
`video/mpeg`	`mpeg mpg mpe`
`video/quicktime`	`qt mov`

1.5 Content Types

On the Web, many different types of files can be placed and retrieved. The Web server and Web browser use a set of standard designations to indicate file content types in order to process different files correctly.

The Web borrowed the content type designations from the Intenet email system and use the same MIME (*Multipurpose Internet Mail Extensions*) defined content types. There are hundreds of content types in use today.

Many popular types are associated with standard file extensions. Table 1.1 gives some examples.

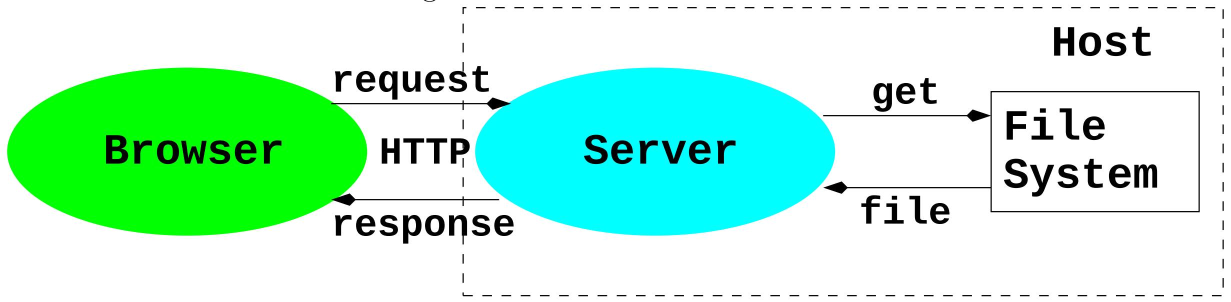
When a Web server returns a document to a browser, the content type is indicated. The content type information allows browsers to decide how to process the incoming content. Normally, HTML, text, GIF, JPEG, PNG etc. are handled by the browser directly. Others types such as quicktime, PDF, audio and video are handled by plug-ins or helper programs.

1.6 Putting Information on the Web

Now let's turn our attention to how information is supplied on the Web. The understanding sheds more light on how the Web works and what it takes to serve up information on the Web.

The Web puts the power of publishing in the hands of anyone with a computer connected

Figure 1.5: Web Server Function



to the Internet. All you need is to run a Web server on this machine and to establish files for it to service. Major computer vendors offer commercial Web servers with their computer systems. Examples are `Windows 2000` (Microsoft), `Solaris/iPlanet` (Sun Microsystems), and `NetWare` (Novell). `Apache` is a very popular Unix-based Web server freely available from `www.apache.org` (the Apache Software Foundation).

Once a Web server is up and running on your machine, all types of files can be served (Figure 1.5) including hypertext (`.html`), plain text (`.txt`), graphical image (e.g. `.gif`), sound (e.g. `.wav`), video (e.g. `.mov`), etc.

On a typical UNIX system, follow these simple steps to make your personal Web page:

1. Make a publicly accessible file directory in your home directory (~*userid*/`public_html`) to contain your files for the Web. This is your *personal Web directory*. Make this directory accessible

   chmod `o+x` ~*userid*/`public_html`

   When in doubt, ask you system mangers about the exact name to use for your personal Web directory.

2. In your Web directory, establish a homepage, usually `index.html`, in HTML. The homepage usually functions as an annotated table of contents. Make this file readable.

   chmod `a+r` ~*userid*/`public_html/index.html`

3. Place files and directories containing desired information in your Web directory. Make each directory and each file accessible as before. Refer to these files with links in the homepage and other pages.

4. Let people know the URL of your homepage which is typically

   `http://`*your-sever*`/~`*your-userid*`/`.

In a Web page, you refer to another file in the same directory with a simple link containing a partial URL (`<a href="`*filename*`">`) where *filename* can be either a simple file name or a pathname relative to the current document.

Among the Web file formats, hypertext is critical because it provides a means for a document to link to other documents.

1.7 What is HTML

A document written in HTML contains ordinary text interspersed with *markup tags* and uses the `.html` filename extension. The tags mark portions of the text as title, section header, paragraph, reference to other documents, etc. Thus, an HTML file consists of two kinds of information: *contents* and HTML tags. A browser follows the HTML tags to layout the page content for display. Because this, line breaks and extra white space between words in the content are mostly ignored. In addition structuring and formatting contents, HTML tags can also reference graphics images, link to other documents, mark reference points, generate forms or questionnaires, and invoke certain programs. Various visual editors or *page makers* are available that provide a GUI environment for creating and designing HTML documents. For substantial website creation projects, it will be helpful to use *Integrated Decelopment Environments* such as Macromedia Dreamweaver (Chapter 11). If you don't have ready access to such tools, a regular text editor can be used to create or edit Web pages. An HTML tag takes the form `<`*tag*`>`. A *begin tag* such as `<h1>` (level-one section header) is

Table 1.2: Some HTML Tags

Marked As	**HTML Tags**
Entire Document	`<html>...</html>`
Header part of document	`<head>...</head>`
Document Title	`<title>...</title>`
Document Content	`<body>...</body>`
Level *n* Heading	`<h`*n*`>...</h`*n*`>`
Paragraph	`<p>...</p>`
Unnumbered List	`<ul>...</ul>`
Numbered List	`<ol>...</ol>`
List Item	`<li> ...</li>`
Comment	`<!-- ... -->`

paired with an *end tag*, `</h1>` in this case, to mark content in between. Table 1.2 lists some frequently used tags.

The following is a sample HTML page (Ex: **Fruits**):

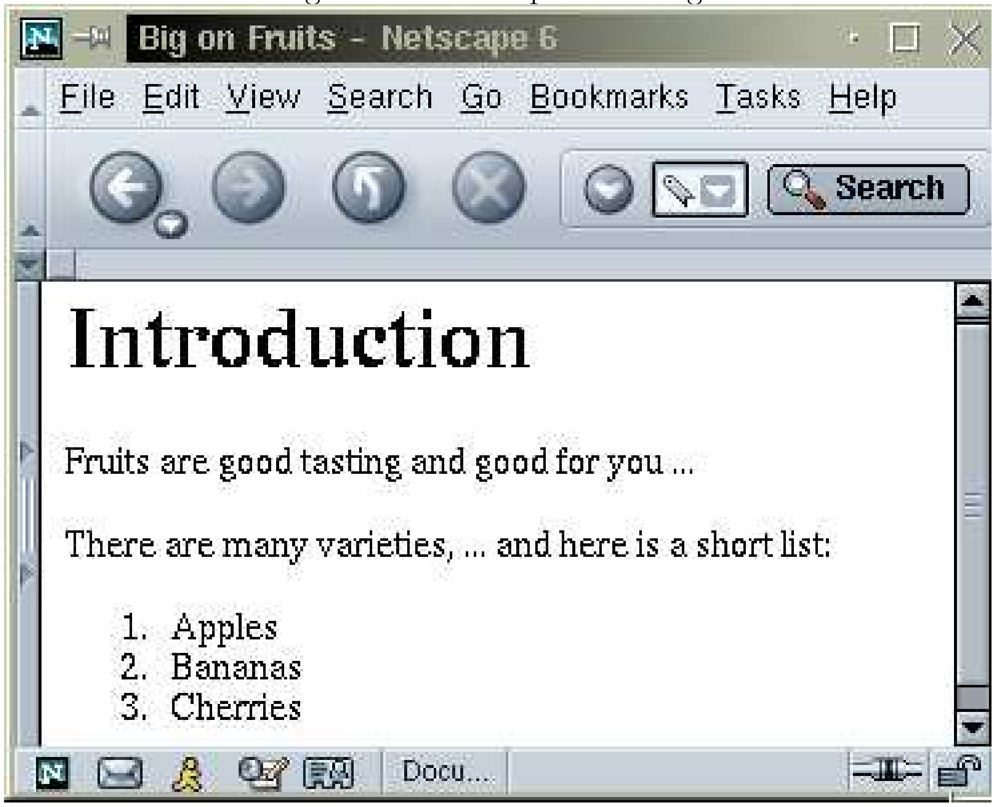
```
<html>
<head> <title>A Basic Web Page</title> </head>
<body>
   <h1>Big on Fruits</h1>
   <p>Fruits are good tasting and good for you ...</p>
   <p> There are many varieties, ...
   and here is a short list: </p>
   <ol>
      <li> Apples </li>
      <li> Bananas </li>
      <li> Cherries </li>
   </ol>
</body></html>
```

Figure 1.6 shows the *Big on Fruits* page displayed by Netscape. We begin in-depth coverage of HTML in Chapter 2.

1.8 Web Hosting

Web Hosting is a service to store and serve ready-made files and programs so they are accessible on the Web. Hence, publishing on the Web involves

Figure 1.6: A Sample Web Page


Big on Fruits - Netscape 6
File Edit View Search Go Bookmarks Tasks Help
Search
Docu...

Introduction

Fruits are good tasting and good for you ...

There are many varieties, ... and here is a short list:

1. Apples
2. Bananas
3. Cherries

1. designing and constructing the pages and writing the programs for a website

2. placing the completed website with a hosting service

Colleges and universities host personal and educational sites for students and faculty without charge. Web hosting companies provides the service for a fee.

Commercial Web hosting can provide secure data centers (buildings), fast and reliable Internet connections, specially tuned Web hosting computers, server programs and utilities, network and system security, daily backup, and technical support. With each hosting account come an amount of disk space, monthly network traffic allowance, email accounts, Web-based site management and maintenance tools, and other access such as FTP and SSH (secure login).

To host a site under a given domain name, a hosting service associates that domain name to an IP number assigned to the hosted site. The domain-to-IP association is made through domain name servers (DNS) managed by the hosting service.

For truly global websites, hosting services such as those provided by AKAMAI can distribute a site in multiple countries for much faster access from anywhere in the world.

1.9 Domain Registration

To obtain a domain name you need the service of a *domain name registrar*. Most will be happy to register your new domain name for a very modest yearly fee. Once registered, the domain name is property belonging to the *registrant*. No one else can register for that particular domain name as long as the current registrant keeps the registration in good order.

ICANN accredits commercial registrars for common TLDs including `.com`, `.net`, `.org`. New TLDs being added include `.biz`, `.info`, `.pro`, `.aero`, `.name`, and `.museum`. Some registrars, such as VeriSign (formerly Network Solutions), also register `.edu` domains. Other restricted domains (e.g. `.gov` and `.us`) are handled by special registries (e.g. `nic.gov` and `nic.us`). Country-code TLDs are normally handled by registries in respective countries.

Accessing Domain Registration Data

The registration record of a domain name is publicly available. The standard Internet *WHOIS* service allows easy access to this information. On UNIX systems easy access to WHOIS is provided by the **whois** command:

whois *domain name*

lists the domain registration record kept at a major *network information center* (nic). For example (Ex: **Whois**),

whois `kent.edu`

produces the following information

```
   Domain Name: KENT.EDU
   Registrar: NETWORK SOLUTIONS, INC.
   Whois Server: whois.networksolutions.com
   Referral URL: http://www.networksolutions.com
   Name Server: NS1.OAR.NET
   Name Server: NS.MCS.KENT.EDU
   Name Server: DHCP.NET.KENT.EDU
   Name Server: NS.NET.KENT.EDU
   Updated Date: 13-jun-2003
```

This record is in summary form. By querying the particular registrar's whois server, a more detailed domain name registration record can be obtained. For example, depending on the computer system used, one of the commands

whois `-h whois.networksolutions.com kent.edu`

whois `kent.edu@whois.networksolutions.com`

will produce the detailed domain record for `kent.edu`:

```
[whois.networksolutions.com]

Registrant:
   Kent State University (KENT-DOM)
```

```
125 Library
Kent, OH 44242
US

Domain Name: KENT.EDU

Administrative Contact, Technical Contact, Billing Contact:
   Yoho, Ransel  (RY678)  ransel@NET.KENT.EDU
   Kent State University
   120 Library
   Kent, Ohio 44242
   330-672-9576 (FAX) 330-672-9593

Record last updated on 21-Jul-2002.
Record created on 19-Feb-1987.
Database last updated on 9-Oct-2003 10:48:00 EDT.

Domain servers in listed order:

NS.NET.KENT.EDU              131.123.1.1
NS.MCS.KENT.EDU              131.123.2.130
DHCP.NET.KENT.EDU            131.123.252.2
NS1.OAR.NET                  192.88.193.144
```

On Linux systems, the WHOIS command are sometimes called **fwhois**.

On Mac OS X, some neat networking tools can be found in the *NetProbe package* including **Ping**, **DNS Lookup**, **Trace IP Route**, **WhoIs**, and **Finger**.

On Windows systems there is no built-in WHOIS program. But you can easily find a freeware **whois** program on the Web (just search for "whois for Windows"), similarly for other common Internet tools.

On-Web whois searches are also available (e.g. `www.crsnic.net` at VeriSign).

1.10 What Are Name Servers

Name servers are the actual programs that provide the domain-to-IP mapping information on the Internet. We mentioned that DNS provides a distributed database service that supports dynamic retrieval of information contained in the name space. Web browsers, and other Internet client applications, will normally use the DNS to obtain the IP a target host before making contact with a server.

There are three elements to the DNS: the name space (Section 1.2), the *name servers*, and the *resolvers*.

- *Name servers*: Information in the distributed DNS are divided into *zones* and each zone is supported by one or more name servers running on different hosts. A zone is associated with a node on the domain tree and covers all or part of the subtree at that node. A name server that has complete information for a particular zone is said to be an *authority* for that zone. Authoritative information is automatically distributed to other name servers which provide redundant service for the same zone. A server relies on lower-level servers for other information within its subdomain and on external servers for other zones in the domain tree. A server associated with the root node of the domain tree is a *root server* and can lead to information anywhere in the DNS. An authoritative server uses local files to store information, to locate key servers within and without its domain, and to cache query results from other servers. A boot file, usually `/etc/named.boot`, configures a name server and its data files.

  The management of each zone is also free to designate the hosts that run the name servers and to make changes in its authoritative database. For example, the host `ns.nic.ddn.mil` may run a root name server. The host `condor.mcs.kent.edu` may run a name server for the domain `mcs.kent.edu`.

  A server answers queries from resolvers and provides either definitive answers or referrals to other name servers. The DNS database is set up to handle network address, mail exchange, host configuration, and other types of queries, some yet to be implemented.

- *Resolvers*: A DNS *resolver* is a program that sends queries to name servers and obtains replies from them. On UNIX systems, a resolver usually takes the form of a C library function. A resolver can access at least one name server and use that name server's information to answer a query directly, or pursue the query using referrals to other name servers.

Resolvers, in the form of networking library routines, are used to translate domain names into actual IP addresses. These library routines, in turn, ask prescribed name servers to resolve the domain names. The name servers to use for any particular host are normally specified in the file `/etc/resolv.conf` or `/usr/etc/resolv.conf`.

The ICANN maintains *root name servers* associated with the root node of the DNS tree. Domain name registrars, corporations, organizations, Web hosting companies, and other *Internet Service Providers* (ISP) run name servers for their zones to associate IPs to domain names in their particular zones. All name servers on the Internet cooperate to perform domain-to-IP mappings on-the-fly.

1.11 Looking up Host Information

On UNIX and MS/Windows systems, the **host**, **dig** and **nslookup** commands provide direct user access to the DNS. These commands are similar but **dig** is intended to be replacing **nslooup**. You may find one or all three work on your system. We'll see how **nslookup** works, because it provides simpler output. The form

nslookup *host*

submits a name server query and obtains domain name, IP address, and alias information for the given host. The name server used is specified in the `resolv.conf` file. To check on the `gopher` server (host) at UICU, for instance, you would enter the command

nslookup `www.kent.edu`

which gives

```
Server:  clmboh1-dns3.columbus.rr.com
Address:  65.24.0.166
```

```
Non-authoritative answer:
Name:    info.cs.kent.edu
Address:  131.123.32.129
Aliases:  www.cs.kent.edu
```

The desired information together with the identity of the name server (from RoadRunner in Columbus Ohio) that provided the data is displayed. As this example shows, typically the name `www` is an DNS alias for a host whose real domain name is something else.

Nslookup is very handy for verifying the existence of hosts and finding the IP address or domain name aliases for hosts. Once the name of a host is known, you can also test if the host is up and running, as far as networking is concerned, with the **ping** command.

ping *host*

This sends a message to the given remote *host* requesting it to respond with an echo if it is alive and well. If this command is not on your command search path, try the command **/etc/ping** or **/usr/etc/ping** instead.

1.12 The Web Development Process

With enough background information, we now turn to our main subject *Web Design and Programming* (WDP). WDP involves conceptualizing, architecting, designing, organizing, implementing, maintaining, and improving websites for functionally effective and esthetically attractive information delivery and exchange.

The Web is a new mass communication medium. To create a well designed and effective website is not easy. It takes expertise in information architecture, visual communication design, mass communication, computer programming, business administration, and consumer psychology, just to name some areas. It usually takes teamwork to get it done.

The heart of the enterprise is a combination of artistic design and computer programming. In this text, we attempt to cover these two areas in an integrated fashion.

To create a website, there are many tasks involved. The overall *website development process* can be summarized here. Subsequent chapters provide in-depth coverage of these tasks.

Requirement Analysis and Development Plan—What are the requirements for the finished site? What exactly will your finished website achieve for the client? What problems does your client want you to solve? Who are the target audiences of the website? Can you realistically help the client? What is the scope and nature of work? What are the design and programming tasks involved? What resources and information will be needed and what problems you foresee? Who will provide content information for the site, in what formats? What resources are needed or available: textual content, photos, imagery, audio, video, logos, corporate identity standards, copyrights, credits, footers, and insignia?

Answer the preceding questions and make a plan, create and group content, functional, and look and feel requirements, and set clear goals and milestones for building and developing the site.

Site Architecture—Decide on an appropriate architecture for the site. The site architecture is influenced by the nature of the information being served and the means of delivery. Ordinary sites involve static pages with text and images and online forms. Specialized sites may involve audio, video, steaming media, and dynamically generated information, or access to databases.

Website *information architecture* (IA) deals with the structuring, the relationship, the connectivity, the logical organization, and the dynamic interactions among the constituent parts of a website.

Within each Web page, consider the placement, layout, visual effect, font and text style, etc. These are also important but may be more "interior decoration" rather than architecture. However, architecture and interior/exterior decoration are intimately related.

The site architecture phase produces a blueprint for building the website. The blueprint is a specification of the components and their contents, functionalities, relations, connectivity, and interactions. Website implementation will follow the architecture closely.

An important aspect of site architecture is the navigation system for visitors to travel in your site. The goal is to establish site-wide (primary), intra-section (secondary), and intra-page (tertiary) navigation schemes that are easy and clear.

Text-only Site Framework—Follow these steps to prepare a skeletal site as a foundation for making adjustments and for further work to complete the site.

- Content: create content list or inventory; prepare content files ready to be included in Web pages.
- Site Map: draw a relationship diagram of all pages to be created for the site, give each page appropriate titles, show page grouping, on-site and off-site links, distinguish static or dynamic pages, identify forms and server-side support. Major subsections of the site can have there own submaps.
- Skeletal site: entry page, homepage, typical subpages and sub-subpages, textual contents (can be in summary form), HTML forms with textual layout and descriptions of server-side support, structure of the file hierarchy for the site, well-defined HTML coding standards for pages.
- Navigation: follow the site architecture and site map to link the pages, use textual navigation links with rough placements (top, left, right, or bottom), avoid dead-end pages and avoid confusing the end-user.

Visual Communication and Artistic Design:

- Design concepts: features, characteristics and look and feel of the site. The design must reflect client identity and site purpose.
- Story boards: simple layout sketches based on text-only site for typical pages, HTML forms, and HTML form response pages; header, footer, margins, naviga-

tion bar, logo, and other graphical elements to support the delivery of content; client feedback and approval of story boards.

- Page layout: (for pages at all levels) content hierarchy and grouping; grids, alignments, constants and variables on the page; placement and size of charts, graphs, illustrations, and photos; creative use of space and variations of font, grid, and color; style options and variations.
- Homepage/entry page: Visuals to support the unique function and purpose of entry to the site and homepage as required by site architecture.

Site Production:

- Page Templates: Create templates for typical pages at all levels. Templates are skeleton files used to make finished pages by inserting text, graphics, and other content at marked places in the templates. In other words, a template is a page frame with the desired design, layout and graphics, ready to receive text, links, photos, and other content. A template page may provide HTML, style sheets, Javascript, `head`, `body`, `meta`, `link`, and `script` tags as well as marked places for page content. Templates enable everyone on the project team to complete pages for the site. Advanced templates may involve dynamic server-side features.
- Prototype pages: Use the templates to complete typical pages in prototype form, test and examine page prototypes, present prototype pages to the client and obtain feedback and approval. Make sure that the layout system has been designed versatile and flexible enough to accommodate potential changes in content.
- Client-side programming: Write scripts for browsers and possibly other Web clients that will be delivered together with Web pages to the client side. Such scripts may include style sheets and Javascripts. Client side programs can make Web pages more interactive and responsive.
- Server-side programming: Write programs for form processing, dynamic page generation, database access, e-business, and e-commerce features. Make sure

these follow the site architecture, user-orientation, and visual design.

- Finished pages: Following page prototypes, add text, graphics, photos, animations, audio and video to templates to produce all pages needed; make final adjustments and fine tuning.

Error checking and validation—Apply page checking tools or services on finished pages to remove spelling errors, broken links, and HTML coding problems. Check page loading times.

Testing—Put the site through its paces, try different browsers from different access locations, debug, fine tune, and check against architecture and requirements.

Deploying—Release the site on the Web, make its URL known, and register the site with search engines.

Documentation—Write down a description of the website, its design and functionalities, its file structure, locations for source files of art and programming, a site maintenance guide.

Maintenance—Continued operation and evolution of the website.

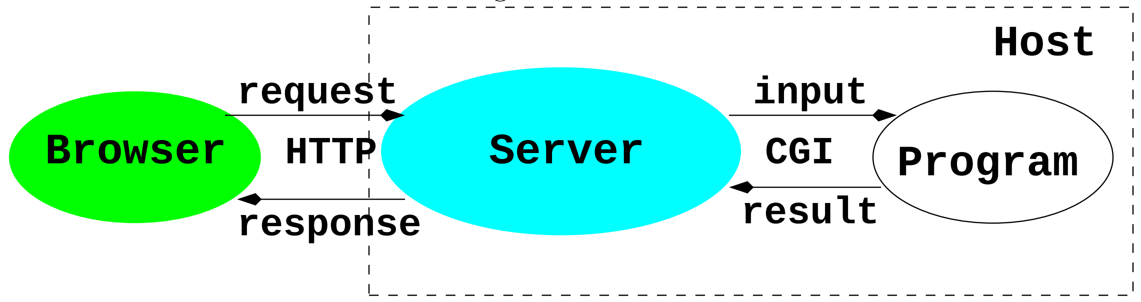
1.13 Dynamic Generation of Web Pages

Documents available on the Web are usually prepared and set in advance to supply some fixed content, either in HTML or in some other format such as plain text, GIF, or JPEG. These fixed documents are *static.* A Web server can also generate documents on-the-fly that bring these and other advantages:

- customizing a document depending on when, where, who, and what program is retrieving it

- collecting user input (with HTML forms) and providing responses to the incoming information

Figure 1.7: CGI



- enforcing certain policies for outgoing documents
- supplying contents such as game scores and stock quotes, that are changing by nature.

Dynamical Web pages are no magic. Instead of retrieving a fixed file, a Web server simply calls another program to compute the document to be returned. As you may have guessed, not every program can be used by a Web server in this manner. There are two ways to add server-side programming:

- Loading programs directly into the Web server to be used whenever the need arises.
- Calling an external program from the server passing arguments to it and receiving results thus generated. Such a program must conform to the *Common Gateway Interface* (CGI) specifications governing how the Web server and the external program interact (Figure 1.7).

Dynamic Server Pages

Dynamic generation of pages is made simpler and more integrated with Web page design and construction by allowing a Web page to contain *active parts* that are treated by the Web server and transformed into desired content on-the-fly as the page is retrieved and returned to a client browser.

The active parts in a page are written in some kind of notation to distinguish them from the static parts of a page. The *ASP* (Active Server Page from Microsoft), *JSP* (Java Server Page), and the popular *PHP* (Hypertext Preprocessor) are examples.

Because active pages are treated by modules loaded into the Web server, the processing is faster and more efficient as compared to CGI programs. Active pages also provide form processing, HTTP sessions, as well as easy access to databases and, therefore, offer complete server-side support for dynamic Web pages.

Both CGI and server pages can be used to support HTML forms (Chapter 8), the familiar fill-out forms you often see on the Web.

1.14 HTTP Briefly

On the Web, browser-server communication follows the HTTP protocol. It is good for a Web developer to have a basic understanding of HTTP. Here is the framework of an HTTP transaction:

1. *Connection*—A browser (client) opens a connection to a server.
2. *Query*—The client requests a resource controlled by the server.
3. *Processing*—The server receives and processes the request.
4. *Response*—The server sends the requested resource back to the client.
5. *Termination*—The transaction is done and the connection is closed unless another transaction will take place immediately between the client and server.

HTTP governs the format of the query and response messages (Figure 1.8). The header part is textual and each line in the header should end in RETURN and NEWLINE but it may end in just NEWLINE.
The initial line identifies the message as a query or a response:

- A query line has three parts, separated by spaces: a *query method* name, a local path of the requested resource, and an HTTP version number. For example,

```
GET   /path/to/file/index.html   HTTP/1.1
```

Figure 1.8: HTTP Query and Response Formats

initial line	(different for query and response)
HeaderKey1: *value1*	(zero or more header fields)
HeaderKey2: *value2*	
HeaderKey3: *value3*	
	(an empty line with no characters)

Optional message body contains query or response data. The amount and type of data in the body are specified in the headers.

or

```
POST   /path/script.cgi   HTTP/1.1
```

The `GET` method simply request the specified resource and does not allow a message body. A `GET` method can invoke a server-side program by specifying the CGI or active-page path, a question mark (`?`), then a *query string*:

```
GET /cgi-bin/newaddr?name=value1&email=value2   HTTP/1.0
```

Unlike `GET`, the `POST` method allows a message body and is designed to work with HTML forms (Chapter 8).

- A response (or status) line also has three parts separated by spaces: an HTTP version number, a status code, and a textual description of the status. Typical status lines are:

```
HTTP/1.0   200   OK
```

for a successful query, or

```
HTTP/1.0   404   Not Found
```

when the requested resource cannot be found.

The HTTP response sends the requested file together with its content type (Section 1.5) and length (optional) so the client will know how to process it.

1.15 Summary

The Web is a versatile and all-encompassing distributed information system. Web servers and clients use the HTTP protocol which is based on TCP/IP.

On the Web resources are identified by URLs. The *host* part of the URL identifies a host computer either by it IP address or by its domain name.

Web pages are usually written in HTML that can markup text and supply hyperlinks using partial and full URLs. Web pages are places under Web servers and become available on the Web. Commercial hosting companies provide hosts and servers to serve Web pages online.

Generating simple Web pages is very easy. But well-designed and implemented websites take skill and training to build. The *Web Design and Development Process* described in Section 1.12) shows the scope and depth of efforts needed for a professionally developed site.

It takes technical understand, talent, and team work to build websites. The most central are design and programming. This book covers both the artistic design and programming aspects of WDP in the subsequent chapters.

Exercises

Review Questions

1. What is a computer network? Name major components in a computer network.

2. What is a networking client? a networking server? a networking protocol?

3. What addressing scheme does the Internet use? What is the format of an IP address? What is the quad notation?

4. Consider the IP address:

   ```
   123.234.345.456
   ```

   Is there anything wrong with it? Please explain.

5. What is DNS? Why do we need it? How are TLDs registered?

6. What do name servers do? Why do we need them?

7. What is the relation between the Web and the Internet? What is the relation between HTTP and TCP/IP?

8. What are the major components of the Web? Why is HTML central to the Web?

9. What is the difference between a Web server and a Web browser? Is the Web server a piece of hardware or software? Explain.

10. How does a Web page get from where it is to the computer screen of an user?

11. What is an URL? What is the general form of an URL? Explain the different URL schemes.

12. What are content types? How are they useful?

13. What are the major tasks in the overall website development process? What knowledge and skills do you expect to learn to be able to participate in this process?

14. What is the difference between a static Web page and a generated Web page?

15. What is an HTTP transaction? A Query? A response?

Assignments

1. Take the domain name `sofpower.com` and write down the full URL that will access its website. Use that URL to visit the site. Find this book's website there.

2. Take the domain name `sofpower.com` and find its IP address. Use this IP address instead of the domain name to visit the site. Write down the bit pattern for this IP address.

3. Search on the Web for the Internet Corporation for Assigned Numbers and Names. Visit the site and discover its mission and services.

4. Find the domain record for `sofpower.com`. Who is the owner of this domain name? Who is the administrative and technical contact?

5. Type the following URL in the `Location` window of your browser:

   `telnet://`*HostDomainName*

   where *HostDomainName* identifies a computer where you have an account and are allowed to login. You should be able to proceed and login.

6. This assignment gives you a real experience with HTTP. Assume that

   `http://{\sl targetHost}\verb`/index.html?

   is a valid URL and the page exists.

   (a) Start the **telnet** application on your computer.

   (b) From the `Terminal` menu, start logging to a file, say `savedfile.html`.

   (c) Connect to *targetHost* at port 80.

   (d) Type on the keyboard exactly and without delay (You won't see your typing on the screen! DO NOT COPY and PASTE.)

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

and hit ENTER or RETURN twice. The **telnet** application will terminate and close automatically.

(e) View `savedfile.html` now with a text editor. What do you get? Remove the HTTP header and save the modified file as a well-formed HTML file.

(f) Now view the `savedfile.html` with a web browser. What do you get?

Can you explain what you have done? (Hint: Section 1.14).