CS-160 Lab #5: Operating Systems & Internet

General

Read through the "Background" material below and the online tutorials, then download the Lab #5 question set and answer the questions. Turn in the questions using the instructions posted on the class web site.

For ALL Word processing documents, you must submit your documents in one of the following formats: MS-Word (NOT Works), RTF (most word processors can save in this format), or Open Document (used by the freely available Open Office suite). They will be returned ungraded if submitted in any other format.

Concepts

This lab will help you review the concepts presented in operating systems module of the online tutorials and adds additional information on key concepts of the Web. By completing this module you will learn about the goals and fundamental components of operating systems, fundamentals of the Web and HTML.

Background

Read and review the operating systems module in the on-line tutorials. Also read the background material on the Internet and the WWW that I have provided below. While reading the online operating systems module, focus your studies on understanding the 4 key ***components*** of an operating system:

1. Process Management (section on processes)
2. Memory Management (sections on memory allocation and virtual memory)
3. I/O Management (sections on resources, synchronization, and deadlocks)
4. File Management (section on file management)

Here is some more detailed background on Internet and HTTP protocol that you will need for this lab:

(From "Introduction to Programming Using Java" David J. Eck. Online book: http://math.hws.edu/javanotes/).

World Wide Web

One of the newest and most interesting Internet developments has been the World Wide Web. Before the Web, individual Internet computers had windowing systems and graphical capabilities, but network tools, like mail, FTP and TELNET, were still text-based. The Web changed that by introducing a graphical, point-and-click network interface. Now you can connect to a Web site, download a graphical page, use your mouse to click on an item of interest, and load another page.

The Web has two main components - the HTML language used to describe web pages, and the HTTP protocol used to transfer HTML across the net. Universal Resource Locators (URLs) are used by both HTML and HTTP to name pages.

HTML Documents

An HTML document describes the contents of a page. A Web browser interprets the HTML code to determine what to display on the page. The HTML code doesn't look much like the resulting page that appears in the browser. The HTML document does contain all the text that appears on the page, but that text is "marked up" with commands that determine the structure and appearance of the text and determine what will appear on the page in addition to the text.

HTML has developed rapidly in the last few years, and it has become a rather complicated language. In this section, I will cover just the basics of the language. While that leaves out all the fancy stuff, it does include just about everything I've used to make the Web pages in this on-line text.

It is possible to write an HTML page using an ordinary text editor, typing in all the mark-up commands by hand. However, there are many Web-authoring programs that make it possible to create Web pages without ever looking at the underlying code. Using these tools, you can compose a Web page in much the same way that you would write a paper with a word processor. For example, Netscape Composer, which is part of Netscape Communicator, works in this way. However, my opinion is that making high-quality Web pages still requires some work with raw HTML, and serious Web authors still need to learn the HTML language.

The mark-up commands used by HTML are called tags. An HTML tag takes the form

<**tag-name optional-modifiers**>

Where the **tag-name** is a word that specifies the command, and the **optional-modifiers**, if present, are used to provide additional information for the command (much like parameters in subroutines). A modifier takes the form

modifier-name = **value**

Usually, the **value** is enclosed in quotes, and it must be if it is more than one word long or if it contains certain special characters. There are a few modifiers which have no value, in which case only the name of the modifier is present. HTML is case insensitive, which means that you can use uppercase and lowercase letters interchangeably in tags and modifiers.

A simple example of a tag is `<HR>`, which draws a line -- also called a "horizontal rule" -- across the page. The `HR` tag can take several possible modifiers such as `WIDTH` and `ALIGN`. For example, the short line just after the heading of this page was produced by the HTML command:

```
<HR align=center width="33%">
```

The `WIDTH` here is specified as `33%` of the available space. It could also be given as a fixed number of pixels. The value for `ALIGN` could be `CENTER`, `LEFT`, or `RIGHT`. A `LEFT` alignment would shove the line to the left side of the page, and a `RIGHT` alignment, to the right side. `WIDTH` and `ALIGN` are optional modifiers. If you leave them out, then their default values will be used. The default for `WIDTH` is 100%, and the default for `ALIGN` is `LEFT`.

Many tags require matching closing tags, which take the form

<**/tag-name**>

For example, the tag `<PRE>` must always have a matching closing tag `</PRE>` later in the document. The tag applies to everything that comes between the opening tag and the closing tag. The `<PRE>` tag tells a Web browser to display everything between the `<PRE>` and the `</PRE>` just as it is formatted in the original HTML source code, including all the spaces and carriage returns. (But tags between `<PRE>` and `</PRE>` are still interpreted by the browser.) "PRE" stands for preformatted text. All of the sample programs in these notes are formatted using the `<PRE>` command.

It is important for you to understand that when you don't use `PRE`, the computer will completely ignore the formatting of the text in the HTML source code. The only thing it pays attention to is the tags. Five blank lines in the source code have no more effect than one blank line or even a single blank space. Outside of `<PRE>`, if you want to force a new line on the Web page, you can use the tag `<BR>`, which stands for "break". For example, I might give my address as:

```
Mitchel Fry<BR/>
```

```
Department of Computer Science<BR/>
Chemeketa Community College<BR/>
Salem, OR 97351<BR/>
```

If you want extra vertical space in your web page, you can use several
's in a row.

Similarly, you need a tag to indicate how the text should be broken up into paragraphs. This is done with the <P> tag, which should be placed at the beginning of every paragraph. The <P> tag has a matching </P>, which should be placed at the end of each paragraph. The closing </P> is technically optional, but it is considered good form to use it. If you want all the lines of the paragraph to be shoved over to the right, you can use <P ALIGN=RIGHT> instead of <P>. (This is mostly useful when used with one short line, or when used with
 to make several short lines.) You can also use <P ALIGN=CENTER> for centered lines.

By the way, if tags like <P> and <HR> have special meanings in HTML, you might wonder how I can get them to appear here on this page. To get certain special characters to appear on the page, you have to use an entity name in the HTML source code. The entity name for < is <, and the entity name for > is >. Entity names begin with & and end with a semicolon. The character & is itself a special character whose entity name is &. There are also entity names for nonstandard characters such as the accented e, é, which has the entity name é.

The rest of this page discusses several other basic HTML tags. This is not meant to be a complete discussion. But it is enough to produce interesting pages.

Overall Document Structure

HTML documents have a standard structure. They begin with <HTML> and end with </HTML>. Between these tags, there are two sections, the head, which is marked off by <HEAD> and </HEAD>, and the body, which -- as I'm sure you have guessed -- is surrounded by <BODY> and </BODY>. Often, the head contains only one item: a title for the document. This title might be shown, for example, in the title bar of a Web browser window. The title should not contain any HTML tags. The body contains the actual page contents that are displayed by the browser. So, an HTML document takes this form:

```
<HTML>

    <HEAD>
        <TITLE>page-title</TITLE>
    </HEAD>

    <BODY>
        page-contents
    </BODY>

</HTML>
```

Web browsers are not very picky about enforcing this structure; you can probably get away with leaving out everything but the actual page contents. But it is good form to follow this structure for your pages.

The <BODY> tag can take a number of modifiers that affect the appearance of the page when it is displayed. The modifier named BGCOLOR can be used to set the background color of the page. For example,

```
<BODY bgcolor=white>
```

will ensure that the background color for the page is white. You can add modifiers to control the color of regular text (TEXT), hypertext links (LINK), and links to pages that have already been visited (VLINK). When the user clicks and holds the mouse button on a link, the link is said to be active; you can control the

color of active links with the ALINK modifier. For example, how about a page with a black background, white text, blue links, red active links, and gray visited links:

```
<BODY bgcolor=black text=white link=blue alink=red vlink=gray>
```

There are several standard color names that you can use in this context, but if you want complete control, you'll have to learn how to specify colors using hexadecimal numbers. It is also possible to use an image for the background of the page, instead of a solid color. Look up the details if you are interested.

Headings and Font Styles

HTML has a number of tags that affect the size and style of displayed text. For a heading, which is meant to stand out on a line by itself, HTML offers the tags <H1>, <H2>, ..., <H6>. These tags are always used with matching closing tags such as </H1>. The <H1> tag is meant for the most important headings and produces the largest size text. I've found <H4> through <H6> to be too small to be useful. You can use
 tags in headings, if you want multi-line headings. You can also use links and images, which are described below. The heading tags can take ALIGN as a modifier, with the value LEFT, RIGHT, or CENTER. For example, the heading

A Sample Heading

was written as "<H1 align=center>A Sample Heading</H1>" in the HTML source code.

There are a number of different style tags that you can apply to text. For example, **bold** text can be obtained by surrounding the text with <B> and </B>. You can use <i> for italic, <U> for underlined, and <TT> for typewriter style text. Most browsers support <SUB> for subscripted text and <SUP> for superscripted text. For example, "x²" will give: x^2.

Because HTML is meant to describe the logical structure of a document, rather than its exact appearance, it has a number of tags for displaying the logical style of the text. For example, the <EM> tag is meant to *emphasize* the text surrounded by <EM> and </EM>, while <STRONG> is for **strong emphasis**. And the <CITE> style tag is meant for *titles of books.*

You can get even more control over the style of the text by using the <FONT>...</FONT> tag. The <FONT> tag uses modifiers such as COLOR and SIZE to control the appearance of the font. For big blue text, you would say:

```
<FONT color=blue size="+1">big blue text</FONT>
```

The value "+1" for the SIZE modifier means "a little bigger than usual." You could use "+2" for an even bigger font, "-1" for a smaller font, and so on. However, only a limited number of different sizes are available.

Lists

There are several tags for producing lists of items. The most widely used of these are <UL> and <OL>. The <OL> tag gives an "ordered list", in which the items are numbered consecutively. The item numbers are provided by the browser. The <UL> tag gives an "unordered list", in which the items are all marked with the same special symbol. In the HTML source code, each list item is indicated by placing a <LI> tag at the beginning of the item. The end of the list is marked by the appropriate closing tag, </OL> or </UL>. For example, the following source code:

```
<UL>
<LI>Isaac Asimov
```

```
<LI>Ursula Leguin
<LI>Greg Bear
<LI>C. J. Cherryh
</UL>
```

produces this list:

- Isaac Asimov
- Ursula Leguin
- Greg Bear
- C. J. Cherryh

Links

The most distinctive feature of HTML is that documents can contain links to other documents. The user can follow links from page to page and in the process visit pages from all over the Internet.

The <A> tag is used to create a link. The text between the <A> and its matching </A> appears on the page. Usually, it is underlined and in a special color. The user can follow the link by clicking on this text. The <A> tag uses the modifier HREF to say which document the link should connect to. The value for HREF must be a URL (Uniform Resource Locator). A URL is a coded set of instructions for finding a document on the Internet. For example, the URL for my own "home page" is

http://math.hws.edu/eck/

To make a link to this page, such as David's Home Page, I would use the HTML source code

```
<A HREF="http://math.hws.edu/eck/">David's Home Page</A>
```

The best place to find URLs is on existing Web pages. Most browsers display the URL for the page you are currently viewing, and they can display the URL of a link if you point to the link with the mouse.

If you are writing an HTML document and you want to make a link to another document that is in the same directory, you can use a relative URL. A relative URL consists of just the name of the file. For example, the page you are now viewing comes from a directory that also contains the other sections in this chapter. For a link to Section 1, which is in a file named s1.html, the relative URL would be just "s1.html", and the complete link would look like

```
<A HREF="s1.html">Section 1</A>
```

There are also relative URLs for linking to files that are in other directories. Using relative URLs is a good idea, since if you use them, you can move a whole collection of files without changing any of the links between them (as long as you don't change the relative locations of the files).

When you type a URL into a Web browser, you can omit the "http://" at the beginning of the URL. However, in an <A> tag in an HTML document, the "http://" can only be omitted if the URL is a relative URL. For a normal URL, it is required.

Images

You can add images to a Web page with the <IMG> tag. (This is a tag that has no matching closing tag.) The actual image must be stored in a separate file from the HTML document. The <IMG> tag has a required modifier, named SRC, to specify the URL of the image file. For most browsers, the image should

be in one of the formats GIF (with a file name ending in ".gif") or JPEG (with a file name ending in ".jpeg" or ".jpg"). A so-called animated gif file actually contains a series of images that the browser will display as an animation. Usually, the image is stored in the same place as the HTML document, and a relative URL is used to specify the image file.

The `<IMG>` tag also has several optional modifiers. It's a good idea to always include the `HEIGHT` and `WIDTH` modifiers, which specify the size of the image in pixels. Some browsers, including Netscape, handle images better if they know in advance how big they are. For browsers that can't display images, you can use the `ALT` modifier to specify a string that will be displayed by the browser in place of the image.

The `ALIGN` modifier can be used to affect the placement of the image. "ALIGN=RIGHT" will shove the image to the right edge of the page, and the text on the page will flow around the image. "ALIGN=LEFT" works similarly. (Unfortunately, "ALIGN=CENTER" doesn't have the meaning you would expect. Browsers treat images as if they are just big characters. Images can occur inside paragraphs, links, and headings, for example. Alignment values of `CENTER`, `TOP`, and `BOTTOM` are used to specify how the image should line up with other characters in a line of text: Should the baseline of the text be at the center, the top, or the bottom of the image? Alignment values of `RIGHT` and `LEFT` were added to HTML later, but they are the most useful values.)

For example, here is HTML code that will place an image from a file named figure1.gif on the page.

```
<IMG SRC="figure1.gif" ALIGN=RIGHT HEIGHT=150
                              WIDTH=100 ALT="Figure 1">
```

The image is 100 pixels wide and 150 pixels high. It will appear on the right edge of the page. If a browser can't display images, it will display the string "Figure 1" instead.

There are many places on the Web where you can get graphics for use on your Web pages. For example, http://www.iconbazaar.com makes a large number of images available. You should, of course, check on the owner's copyright policy before using someone else's images on your pages.

HTTP Protocol Overview

(HTTP Protocol Overview from: http://www.freesoft.org/CIE/Topics/102.htm)

The HyperText Transfer Protocol (HTTP) is the de facto standard for transferring World Wide Web documents, although it is designed to be extensible to almost any document format.

HTTP operates over TCP connections, usually to port 80, though this can be overridden and another port used. After a successful connection, the client transmits a request message to the server, which sends a reply message back. HTTP messages are human-readable, and an HTTP server can be manually operated with a command such as `telnet server 80`.

The simplest HTTP message is "`GET` *url*", to which the server replies by sending the named document. If the document doesn't exist, the server will probably send an HTML-encoded message stating this. I say *probably*, because this simple method offers poor error handling and has been deprecated in favor of the more elaborate scheme outlined below.

A complete HTTP 1.0 message begins "`GET` *url* `HTTP/1.0`". The addition of the third field indicates that full headers are being used. The client may then send additional header fields, one per line, terminating the message with a blank link. The server replies in a similar vein, first with a series of header lines, then a blank line, then the document proper.

Here a sample HTTP 1.0 exchange:

```
GET / HTTP/1.0    >
                  >
                     < HTTP/1.0 200 OK
                     < Date: Wed, 18 Sep 1996 20:18:59 GMT
                     < Server: Apache/1.0.0
                     < Content-type: text/html
                     < Content-length: 1579
                     < Last-modified: Mon, 22 Jul 1996 22:23:34 GMT
                     <
                     < HTML document
```

The use of full headers is preferred for several reasons:

- The first line of a server header includes a response code indicating the success or failure of the operation.
- One of the server header fields will be `Content-type:`, which specifies a MIME type to describe how the document should be interpreted.
- If the document has moved, the server can specify its new location with a `Location:` field, allowing the client to transparently retry the request using the new URL.
- The `Authorization:` and `WWW-Authenticate:` fields allow access controls to be placed on Web documents.
- The `Referer:` field allows the client to tell the server the URL of the document that triggered this request, permitting savvy servers to trace clients through a series of requests.

In addition to `GET` requests, clients can also send `HEAD` and `POST` requests, of which `POST`s are the most important. `POST`s are used for HTML forms and other operations that require the client to transmit a block of data to the server. After sending the header and the blank line, the client transmits the data. The header must have included a `Content-Length:` field, which permits the server to determine when all the data has been received.

Uniform Resource Locators

Uniform Resource Locators (URLs) are strings that specify how to access network resources, such as HTML documents. They are part of the more general class of Universal Resource Identifiers (URIs). The most important use of URLs is in HTML documents to identify the targets of hyperlinks. When using a Web browser such as Netscape, every highlighted region has a URL associated with it, which is accessed when the link is activated by a mouse click. *Relative URLs* specify only a portion of the full URL - the missing information is inferred though the context of the source document.

URLs are documented in RFC 1738. Relative URLs are documented in RFC 1808. URIs are documented in RFC 1630.

Here is a URL that describes the root page of the Internet Encyclopedia:

```
http://www.FreeSoft.org/Connected/index.shtml
```

The meaning of these fields is as follows:

`http`

The HyperText Transfer Protocol (HTTP) is to be used to retrieve the document. Other possible values for this field include `https` (use secure HTTP), `ftp` (use the File Transfer Protocol), and `gopher` (use the Gopher Protocol), among others.

`www.FreeSoft.org`

This is a hostname to be resolved using the Domain Name Service

`/Connected/index.shtml`

A directory and filename, to be passed along in the HTTP request to identify the document among many other on the server.