Creating a Web Page on bigmazzy

Your web page is located at the following URL:

http://www.math.uaa.alaska.edu/~username

To create a web page, you must log on to bigmazzy.math.uaa.alaska.edu and create a public_html directory in your home directory. The file index.html will be accessed when someone goes to the address above; otherwise, any other files in the public_html directory can be accessed by appending a "/filename" to the above address. You may already have a public_html directory created for you.

You can either edit the files in this directory by logging into bigmazzy and using an editor such as pico, or you can transfer files there from another machine (e.g. via WinSCP).

You must also change the permissions on your home directory and public_html directory. On the command line, type 'chmod a+x ~' and 'chmod a+x ~/public_html'. Also make sure any files in public_html are readable by everyone;
'chmod -R a+r ~/public_html'. You should be ready to go!

HTML Primer

This HTML primer assumes that you are already familiar with the basics of creating an HTML page. The basics include the following tags:

- TITLE <TITLE>
- Heading <H1> <H2> etc.
- Paragraph Mark <P>
- Horizontal Rule <HR>
- Boldface <B>, Italic <I>, Underline <U>
- Font Size <FONT SIZE=value>
- Font Color <FONT COLOR=#color>
- Image <IMG SRC="reference">
- Bulleted List <UL>, <LI>
- Ordered List <OL>, <LI>
- Background Image <BODY BACKGROUND="image">
- Background Color <BODY BGCOLOR=#color>

For a primer on these and other tags, I am referring students to Dave's Site located at http://www.davesite.com/webstation/html/. There are many other references on the web for HTML, I suggest just starting googling for HTML links if you want further information.

In this primer we will cover just the following additional HTML tags and technologies:

- Forms
- Tables
- Frames
- Button

Forms

World Wide Web "forms" (or "fill-out forms") are the computer equivalant of paper forms we fill out all the time in everyday life, such as an application form. You will find a button or link at the end of every WWW form, often labeled "Submit". When you push this button, two things are sent to the server: the data you've typed into the form, and an ACTION, which basically tells the server the name of the program that knows how to process that form's data. The server simply invokes that program and passes the form's data to it, and arranges for the output of that program to be sent back to the browser (ordinarily an HTML stream).

Common uses of forms are surveys, on-line order forms, feedback, or really any Web page in which input is required from the user in order to accomplish a given task or provide a service to the user.

The FORM tag:

<FORM NAME=”[name of form]” METHOD=POST ACTION="[name of program]">
</FORM>

These tags start and end a form (all input fields of the form are placed between these two tags). METHOD specifies which technical protocol the web server will use to pass the form data to the program which processes it (POST is the most common type, although GET is also a valid type), and ACTION tells the server exactly which program that is.

A web page (i.e., an HTML document) may include multiple forms; however, forms may not be nested (you cannot create a form within a form). You must end the current form with < /FORM > before you can start a new form.

Form Elements:

Submit Box: This sends in the data to the program specified by ACTION.
<INPUT TYPE=submit VALUE="Send in the data">

Reset Box: Clears values back to the defaults.
<INPUT TYPE=reset VALUE=”clear”>

Input Box: This is a single box for text input. You can specify the size and default value.
<INPUT TYPE=text NAME=your_name SIZE=50 VALUE="Joe Schmo">

TextArea : Multi-line input box.
<TEXTAREA NAME="data" ROWS=2 COLS=60></TEXTAREA>

Checkbox: Boxes the user can check or uncheck. The value is the variable that will be sent to the CGI program when the box is submitted and checked. CHECKED indicates the default value.

<INPUT TYPE=checkbox NAME="baseball" VALUE="Y" CHECKED>Baseball

<INPUT TYPE=checkbox NAME="basketball" VALUE="Y">Basketball

Radio Button: Like a checkbox, but use if only one item is selectable.

<INPUT TYPE="radio" NAME="sport" VALUE="soccer" CHECKED>Soccer

<INPUT TYPE="radio" NAME="sport" VALUE="baseball">Baseball

<INPUT TYPE="radio" NAME="sport" VALUE="basketball">Basketball

Option List: A drop-down list of choices. If SIZE is large enough to hold all the items, then all will be displayed. If size is left out, size=1 by default.
<SELECT NAME=”platforms” SIZE=2>
<OPTION>Windows
<OPTION>Macintosh
<OPTION>UNIX
</SELECT>

Password: This is an input box, but the echo will be *’s.
<INPUT NAME=”varName” TYPE=password SIZE=8>

Hidden: This is an invisible value that you might use for your program’s internal processing, for example, a flag indicating the source of this particular web page if you have many web pages sharing a single CGI program.
<INPUT NAME=”varName” TYPE=hidden VALUE="Hidden Value">

Tables

Tables are a staple of today’s web pages, since they can be used to control with a fair degree of precision where text goes on the page. Tables are used extensively on many web sites to bring structure to a page. There are small tables and big tables- small ones are often used simply to encompass and organize a section of the page, such as a chart or spreadsheet-like data; large ones are used to bring organization to the entire document (as seen in the front page of Yahoo).

To create a HTML table, we have to first look at the basic tags necessary to create one:

Tag	Function
<table></table>	Defines a table
<tr></tr>	Defines a row within the table
<td></td>	Defines a cell within the table

Every single table you construct will consist of the above three tags, no matter the complexity.

The following creates an extremely simple table with four cells:

```
<table>
<tr>
<td>Cell #1</td>
<td>Cell #2</td>
<tr>
<td>Cell #3</td>
<td>Cell #4</td>
</tr>
</table>
```

You can add any other tag within a table cell, for example, images.

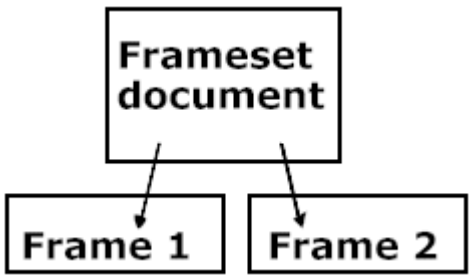
The following are table attributes that are useful:

Attribute	Function
border=?	Specifies the border width of the table, in pixels.
width=?	Specifies the width of the table or cell, in pixels or %.
height=?	Specifies the height of the. table or cell, in pixels or %
cellpadding=?	Specifies the distance between the cell and the content
cellspacing=?	Specifies the spacing between each cells, in pixels.

Frames

Frames are individual windows contained within the main window, each capable of containing a separate document. A nicely designed frames page can help tremendously in site navigation and user interaction, but can be a nightmare for the user if used without care.

The basic concept of a frame document is this: It contains at least three parts- The Frameset document, which holds all of the frames, and at least two frames:



Think of the frameset document as a container that contains two or more separate documents (Frame 1 and Frame 2).

Here is a sample frame that demonstrates the process:

```
<html>
<head>
<title>Homepage</title>
</head>
<frameset cols="10%,90%">
<frame name=frame1 src="frame1.html">
<frameset rows="50%,50%">
<frame name=frame2 src="frame2.html">
<frame name=frame3 src="frame3.html">
</frameset>
</frameset>
</html>
```

Frame1.html contains:

```
This is frame 1!
<p>
<a href="http://www.yahoo.com" target=frame2>Link from frame
1.</a>
```

Frame2.html contains:

```
This is frame 2!
<p>
<a href="http://www.yahoo.com">Link from frame 2.</a>
```

Frame3.html contains:

```
This is frame 3!
<p>
<a href="http://www.yahoo.com">Link from frame 3.</a>
```

The order we specify the framesets is important. We first create two columns, one 10% wide and the other 90% wide. The first one on the left is named Frame1 and will be filled with the html in Frame1.html. The second column on the right is now split into two frames by row, each 50% of the remaining space. Each is populated with their respective html files.

By specifying a name for the TARGET in the HREF tag, we can control which frame the source will load in. If the given frame doesn't exist, a new window will be created.

Button

We already know how to create buttons with forms. The process is:

```
<form>
<input type="button" value="CNN"
onClick="window.location='http://www.cnn.com'">
</form>
```

This is fine, but the button is kind of generic. We can spruce it up a bit using the (HTML 4) button tags.

The format is simply:

<button>Some text here</button>

However, you are free to put a variety of tags inside the button. For example:

Fonts:
<button><big>Big text</big></button>

Pictures:
<button><img src="kitty.gif"></button>

Even tables:
<button>
<table border="1" width="100%">
<tr>
<td width="19%"><img src="foo.gif"</td>
<td width="81%">Under Construction</td>
</tr>
</table>
</button>

The things you can't put inside a button are other buttons, FORM tags, or <a> tags.

To make the button do something useful, you need to add some additional code.
For the most part, adding functionality to the new <button> tag of HTML 4.0 means using JavaScript and the JavaScript onClick event handler.

The below illustrates some possible uses for the button:

Taking surfers to a url
<button onClick="window.location='http://www.uaa.alaska.edu'">Home</button>

Closing a window:
<button onClick="window.close()">Close</button>

Opening a new window with the specified document:
<button onClick="window.open('mydocument.htm')">Open second window!</button>