HTML 2

Using Fonts and Tables

Web Design Series - Session 2

CAL People and Computer Training
University of California, Berkeley

For more information about the CAL PACT program, to sign up for classes, or to download course documentation, please visit our website at: **http://calpact.berkeley.edu/**

Last Revised March 2001

Use this space for notes

Introduction

Welcome to **HTML 2**! If you're here today, you've already completed *HTML1*, which means you're ready to expand your HTML skills. Today we'll continue developing our website for the Bear Net-Works department.

Skills you need for this class

- Text editing
- How to use the mouse
- Familiarity with the Windows or Mac operating systems
- Familiarity with the Internet
- Experience using web browsers, such as Netscape Navigator and Microsoft Internet Explorer
- Understanding of the material covered in *HTML 1*

Skills and concepts you will learn in this class

- Fonts — style, size and color
- Tables — the basics

Conventions used in this document

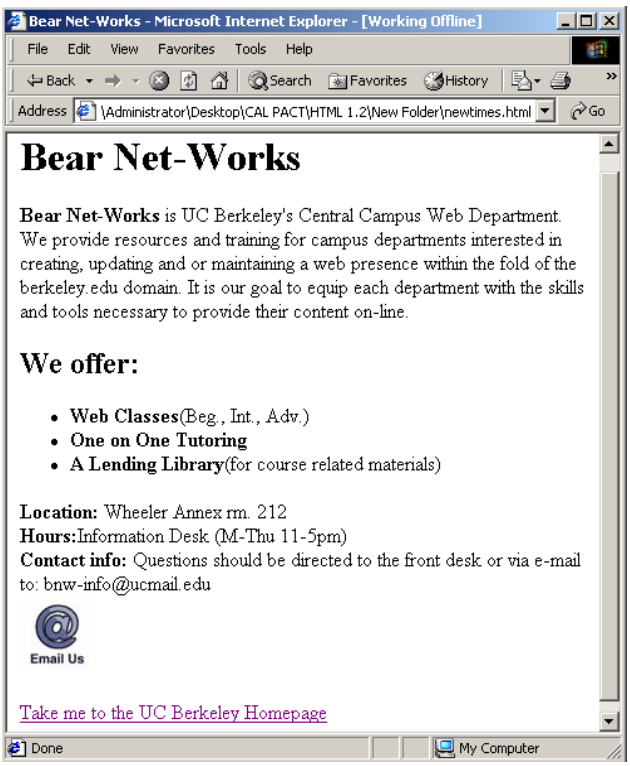
Menus and menu commands are separated by a vertical bar (**|**). In the document they will appear as **Menu|Command**. An example of this is: "Select **File|New...**."

Introduction

Today in **HTML 2**, things will be a bit from our time together last week. We won't be able to include all of the new tags we learn today in our test page. Because of this, new code will be presented by a series of examples. We'll work through them together, and then you will have time to test your skills in practice exercises. In *HTML 1* we learned how to create a webpage using a few basic HTML tags. Today, we'll expand on those in order to change the color, size and style of our text. We'll also learn to use tables—a very simple, yet powerful design tool.

Your instructor should now direct you to our HTML document template (it's our *index.html* file). It contains the code you worked with in HTML 1. Save a copy of the file to the desktop. Next open it in a text editor. Then open that same file in a web browser.

Here's what your browser window should display:



Fonts and Design

The appearance of your text is one of the most important parts of presenting your content. Color, size and style can all affect the impression a reader will form when they visit your site. It is important to make that impression a good one.

Of course, there is plenty of room for individual taste, but there are a few useful rules of thumb. For example, using larger and bolder text for the titles of sections makes it easier for a reader to skim for the information they are looking for.

Note

When deciding on how best to present your content, imagine having to read it yourself. What do you want to stand out? What is your eye drawn to? If you find it difficult to concentrate on one element of the page at a time, what might be distracting you?

You read a lot more than you realize in a given day. What types of text are easier for you to concentrate on: newspaper type, novels, spreadsheets? What about the way these types of text appear helps you as a reader?

The <FONT> Tag and its Attributes

Without specifying in our code which font we would like the browser to display, the browser will resort to its default font setting. The default setting includes values for font, point size, color, and style. To change these values to your needs as a web author, use the <FONT> tag in conjunction with one or several of its attributes. You can change the appearance of any amount of text, provided it falls between a pair of opening and closing **FONT** tags, with the appropriate attributes.

The Typeface Attribute: FACE="*font*"

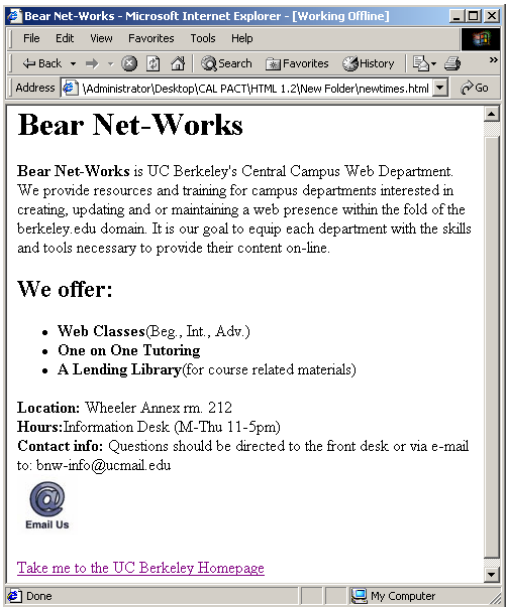
Suppose we want to change the font for our entire document to Arial. To effect the text of the entire document body, place the <FONT> tag just after the opening <BODY> tag, like so: <FONT face="Arial">. Then place the closing </FONT> tag just before the closing </BODY> tag.

Go ahead and edit the template document you just saved to the desktop. Your code should look like this:

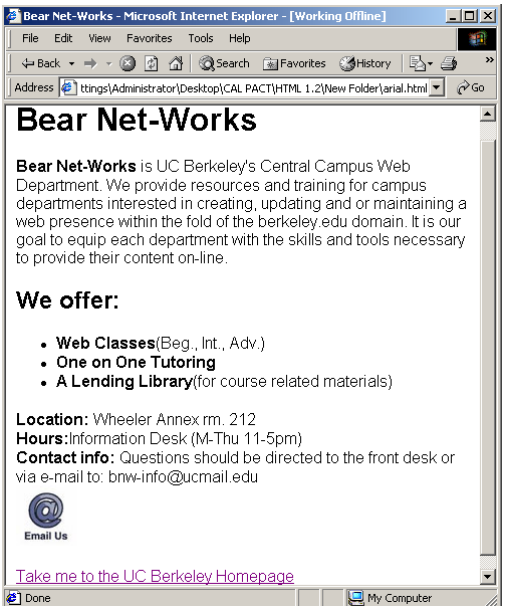
```
<HTML>
<HEAD><TITLE>Bear Net-Works</TITLE></HEAD>
<BODY>
<FONT face="Arial">

...body of our document...

</FONT>
</BODY>
</HTML>
```

By now, you should be somewhat familiar with attributes. Here, the **FACE** attribute is used to indicate which font to use. The font desired is placed between the quotation marks. Values assigned to attributes are not case sensitive, so Arial is equivalent to "aRIAL" and "aRiAl". However, make sure to spell the name of your font correctly. This attribute will tell the browser to look in the user's computer for the font indicated. Keep in mind that the user must have the font installed for this to work correctly.

Our original document



After changing our font to Arial



After saving your changes and reloading the document in the browser window, you should notice the font change. Upon closer inspection you may also notice the ***size*** of the font has changed. This is because different fonts vary slightly in size.

When selecting a font, try to choose one that visitors to your page likely have installed on their computer. But be careful: by placing the name of the font in your HTML code does not guarantee it will be displayed. If you specify a font that isn't installed on a visitor's computer, their browser will instead display the default font. The impact a change of font may have on your presentation will vary, so it's important to think about how strongly the look and feel of your page depends on the font you have chosen.

Because of this, it is always safest to choose from the most commonly installed fonts. Below is a list of fonts most computers are likely to have:

Times New Roman
Verdana
□□□□□□□
Arial
Sans serif

Note

As a rule of thumb, it is best to stick with a single font color, type and size throughout your document, deviating only occasionally to place emphasis on a particular piece of information. Many different fonts, sizes and colors tend to be distracting to a reader.

Using Multiple Fonts

To use multiple fonts in a single document you need only use a second pair of <FONT> tags. Your document text will respond to the closest pair of <FONT> tags it is between. For example:

This code will display	**like so...**
<FONT face="courier">Courier	
<FONT face="arial">Arial</FONT>	▫▫▫▫▫▫▫Arial Verdana
<FONT face="verdana">Verdana</FONT>	
</FONT>	

Note

A browser's default font size is typically 12 point Times New Roman. So font sizes *typically* reflect the following:

1 = 8 point
2 = 10 point
3 = 12 point
4 = 14 point
5 = 18 point
6 = 24 point
7 = 36 point

If, however, a browser's default size has been set up to 14, the font point sizes will go up accordingly.

As you may remember from HTML1, placing tags inside one another is referred to as "nesting".

Font Size Attributes: SIZE="*n*" and POINT-SIZE="*n*"

We can change the size of a font by using either the SIZE or POINT-SIZE attributes with the <FONT> tag.

Using the SIZE attribute, there are two ways to specify the size of a font. We can do it **absolutely** with a number between 1 and 7:

```
<FONT size="3">text goes here</FONT>
```

...or we can do it **relatively** with a number between -7 and +7:

```
<FONT size="+1">text goes here</FONT>
```

Beware

Using relative font sizes is not always the best choice. As you continue to learn more tags, you will notice how tricky it can become to have your page display in exactly the manner you would like. HTML is not a precise language. There are many factors beyond your control that contribute to how your page will display.

What's the difference? When a browser program has been freshly installed, the default font size is set to 3. The user, however, is able to change the default setting to their liking. Some may wish to use a smaller font, others, a larger font.

If you assume a visitor to your page is using the default font size of 3, you may set the title of each new section (absolutely) to a size of 4 or 5 to insure they stand out. However, if a visitor to your site has set the default font size to 4, setting your new sections titles (absolutely) to 4 will not have the desired effect. The fonts will appear the same size as the body of your text. If you were to use ***relative*** settings, you could assign your section titles a size of +1. This would insure that they appeared exactly one size larger than the default font size, no matter what a visitor has their browser set to.

FYI

You can also use the BASEFONT tag to apply the same font size for all the body text on a given page. Use the SIZE attribute an define a value from 1 to 7. A closing tag isn't required.

When using this method, you can still change individual sections of text with the standard FONT tag. The BASEFONT tag does not affect headers.

Using the POINT-SIZE attribute is another option. This will give you absolute control over the size of text on the page. However, this attribute does not work with all browsers and all platforms. (It works best on a Mac with Netscape Navigator). For information on this attribute, refer to an HTML resource on-line or in print.

Bear in mind that different browsers may display text or colors differently. The same applies to PC and Mac platforms. In addition, each browser application comes with a host of customizable features enabling the user to tailor the way in which their web pages are displayed. When creating simple web pages, these differences are not so noticeable; however, as our design and layout become more complex, we are increasingly dependent on factors we are only loosely in control of to determine our design and layout (e.g. font size, line spacing, color, etc.).

Therefore, the more tags we assign relative values to, the less control we give ourselves. For example, if we wish to display an image with a given font in a very specific position on our page, we may find ourselves forced to use a font size of 2 in order for the alignment to be correct. Using a relative font size of -1 (assuming the default is 3) would not be wise in this case. A visitor with a default setting other than 3 would not see the page as we had intended for it to be seen.

Practice Exercise

Format some text in our Bear Net-Works page by using the <FONT> tag with the SIZE or POINT-SIZE attribute with a value of your choosing.

The Color Attribute: COLOR=*"n"*

You can change the color of any font by using the COLOR attribute. There are two ways with which to select a specific color. You may either type the name of the color between the quotation marks, as follows:

```
<FONT color="red">text goes here</FONT>
```

...or you can specify a hexadecimal value:

```
<FONT color="#FF0000">text goes here</FONT>
```

#FF0000 = red

An in-depth discussion of hexadecimal values is beyond the scope of this class. However, there are several places on the Web were you can find a list of colors with their corresponding hexadecimal values. One such place is:

http://htmlgoodies.com/tutors/basic_cl.html

Below is a list of basic colors you may wish to try:

Red	#FF0000	**Blue**	#0000FF	**Brown**	#A52A2A
Orange	#FFA500	**White**	#FFFFFF	**Gray**	#BEBEBE
Yellow	#FFFF00	**Green**	#008000	**Purple**	#800080

Why use hexadecimal values over names? Hexadecimal values give you more specific control over the appearance of your colors. We'll discuss colors in more depth in the *Web Design and Usability* class.

Practice Exercise

Find a word in the Bear Net-Works page and use the COLOR attribute with the FONT tag to change each letter of the word to a different color.

Tables — Tags, Attributes & How to Use Them

HTML tags for creating tables were originally designed for displaying tabular data on a web page (in the form of rows and columns). Crafty web designers soon learn that they could use them to control the layout of their pages. So while you can still use them to display tabular data, their real power is as a design tool. You can create columns of text, place white space between elements, and, in general, control the position of your content with far more accuracy than most HTML tags allow.

Take a look at the following example of an HTML table. To the left is the code needed to create the table. To the right is what a browser will display.

```
<TABLE>
   <TR>
        <TD>Row 1 Column 1</TD>
        <TD>Row 1 Column 2</TD>
    </TR>
    <TR>
        <TD>Row 2 Column 1</TD>
        <TD></TD>
        <TD>Row 2 Column 3</TD>
    </TR>
    <TR>
        <TD>Row 3 Column 1</TD>
       <TD>Row 3  Column 2</TD>
    </TR>
</TABLE>
```

Row 1 Column 1 Row 1 Column 2

Row 2 Column 1 Row 2 Column 3

Row 3 Column 1 Row3 Column 2

The <TABLE> tag is used to define the beginning and end of a table. The opening and closing <TR> tags define each row. Within each row (i.e., between each pair of <TR> tags), the opening and closing <TD> tags define each cell in that row, essentially forming columns if coded correctly. All the information (text, images, etc.) to be included in a table must be placed between the <TD> tags.

Notice we did not place text between the opening and closing <TD> tags in Row 2 Column 2. Doing this allows us to create an empty cell. Keeping this in mind, you may wonder why both Row 1 and Row 2 also have empty cells, despite the fact that we did not define them. How is this possible?

The number of columns in your table is determined by the row with the greatest number of defined columns (cells). Therefore, because Row 2 has three columns, and three columns is the maximum number in any of our rows, our table is displayed with three total columns. Empty space is created when the browser finds no <TD> tags in this case. The cells don't exist, the space is there simply because of the number of cells defined in Row 2.

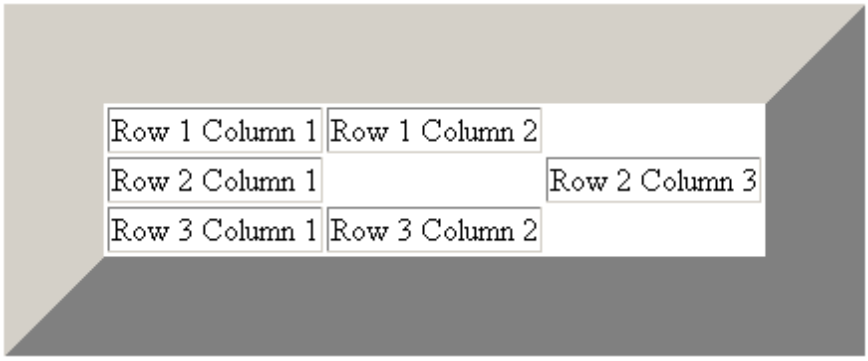
If we wanted to place content in the currently undefined Row 1 Column 3 or Row 3 Column 3, we would have to define the appropriate cell(s) using <TD> tags. The content would then be placed between these tags. Let's add these two sets of <TD> tags now to clean up our table. In a moment we'll discuss placeholders to create blank cells.

The Borders Attribute: BORDER="*n*"

By default, the <TABLE> tag creates a table with no border. While this is preferred when creating page layouts, sometimes a border is desired. To create a border, just

```
<TABLE border="50">
...your table...
</TABLE>
```

add the BORDER attribute to the <TABLE> tag. For example, this code displays as follows...



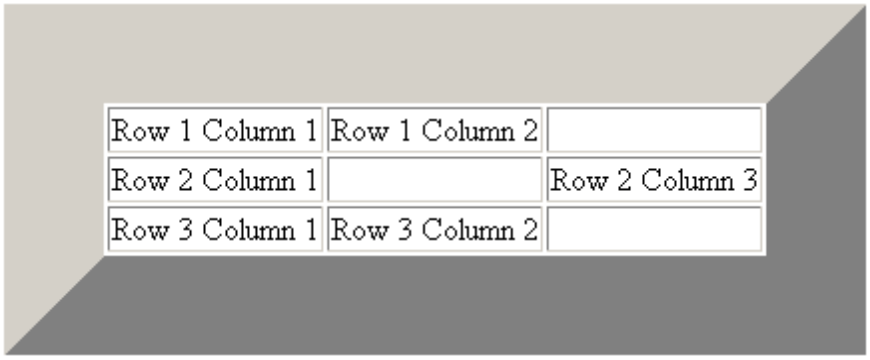
Notice the BORDER attribute created an outer border of 50 pixels in size. This also creates borders inside the table. The cells that contain content are given a default border of two pixels, while the empty cells have no border at all. Sometimes, it may be aesthetically unpleasing to have borders around only some of the individual cells. Fixing this requires a simple HTML trick.

For the empty cells, add the **
** tag between the opening and closing <TD> tags, so the code for the cell reads as follows...

```
<TD><BR></TD>
```

Inserting a line break will fool the browser into thinking there is content in the cell. (This can also be done by using a non-breaking space rather than a line break. This requires a special code, which is beyond the scope of our class today, but which we'll learn more about later in the series. Suffice it to say that if you want to do this, enter ** ** between the <TD> tags.)

After adding this to our code, our new table should look like this:



The Cell Padding Attribute: CELLPADDING="*n*"

Cell padding in a table determines the amount of empty space ***surrounding*** the content within each individual cell. The attribute is written as CELLPADDING="*n*", where *n* is the number of pixels. The default is 1 pixel. Below are a few examples of cell padding values.

cellpadding="0"

Row 1 Column 1

cellpadding="5"

Row 1 Column 1

cellpadding="50"

Row 1 Column 1

The Cell Spacing Attribute: CELLSPACING="*n*"

Cell spacing determines the amount of empty or white space ***between*** individual cells. The attribute is defined by CELLSPACING="*n*" where *n* is the number of pixels. The default is 2 pixels. Compare the sample below with the previous example shown for cell padding to see and understand the difference.

cellspacing="0"

Row 1 Column 1	Row 1 Column 2
Row 2 Column 1	Row 2 Column 2
Row 3 Column 1	Row 3 Column 2

cellspacing="50"

Row 1 Column 1	Row 1 Column 2
Row 2 Column 1	Row 2 Column 2
Row 3 Column 1	Row 3 Column 2

The cell padding examples shown here are used in conjunction with the BORDER attribute. Thus a value of "0" doesn't completely eliminate the inside border. If you are using tables as a design tool, you can completely eliminate this inside border by defining the CELLPADDING attribute with a value of 0 pixels.

The Width and Height Attributes: WIDTH="*n*"; Height="*n*"

When a table is created, a minimum size is used to display the objects in the cell. An exact or relative size can be specified with the WIDTH="*n*" and HEIGHT="*n*" attributes. The value *n* can be an **absolute** number representing the number if pixels, or it can be a **relative** number in the form of a percentage value. This percentage is relative to the size of the web browser window as set by the user. If the size of the browser window is changed, the size of the table is changed accordingly (although there are limits). With an absolute setting, the table will always be displayed at a specific size, regardless of the size of the window. If the browser window is not large enough, scroll bars will appear. Absolute and relative values may be mixed if desired, but use caution when doing so to avoid problems.

Note

The WIDTH and HEIGHT attributes can also be used in conjuction with the <TD> tag for even more control. (They are not used with the <TR> tags, however.)

When doing this, be sure to maintain consistency between table rows.

The following are samples of how these attributes are implemented within the <TABLE> tag.

<TABLE width="300" height="500">	*Creates a 300x500 pixel table*
<TABLE width="75%" height="200">	*Creates a table which occupies 75% of the width of the browser window and is 200 pixels high.*

The Alignment Attributes: ALIGN="*n*"; VALIGN="*n*"

The ALIGN and VALIGN attributes control position. When used with the <TABLE> tag, it controls the position of the table in the browser window. When used with the <TR> tag, it aligns the content inside the cells of an entire row. And when used with the <TD> tag, it aligns the content in a single cell.

The syntax for these attributes is as follows:

for ***horizontal*** alignment:

ALIGN="LEFT/CENTER/RIGHT"

for ***vertical*** alignment:

VALIGN="TOP/CENTER/BOTTOM"

```
<TABLE width="50%" height="50%" border="1" align="center">
    <TR valign="top">
        <TD align=left>Left Top</TD>
        <TD align=center>Center Top</TD>
        <TD align=right>Right Top</TD>
    </TR>
    <TR valign="center">
        <TD align="left">Left Center</TD>
        <TD align="center">Center Center</TD>
        <TD align="right">Right Center</TD>
    </TR>
    <TR valign="bottom">
        <TD align="left"> Left Bottom</TD>
        <TD align="center">Center Bottom</TD>
        <TD align="right">Right Bottom</TD>
    </TR>
</TABLE>
```

The code above displays the following:

Left Top	Center Top	Right Top
Left Center	Center Center	Right Center
Left Bottom	Center Bottom	Right Bottom

The Spanning Attributes: ROWSPAN="*n*" COLSPAN="*n*"

Spanning is a feature that allows a single cell to occupy space in more than one row or column. Look at the example code below.

```
<HTML>
<TABLE border="1">
    <TR>
        <TD rowspan="2">Row 1 and 2 Column 1</TD>
        <TD>Row 1 Column 2</TD>
        <TD>Row 1 Column 3</TD>
    </TR>
    <TR>
        <TD colspan="2">Row 2 Column 2 and 3<BR></TD>
    </TR>
    <TR>
        <TD>Row 3 Column 1</TD>
        <TD>Row 3 Column 2</TD>
        <TD>Row 3 Column 2</TD>
    </TR>
</TABLE>
</HTML>
```

Here is how it would look...

<table>
<tr><td rowspan="2">Row 1 and 2 Column 1</td><td>Row 1 Column 2</td><td>Row 1 Column 3</td></tr>
<tr><td colspan="2">Row 2 Column 2 and 3</td></tr>
<tr><td>Row 3 Column 1</td><td>Row 3 Column 2</td><td>Row 3 Column 2</td></tr>
</table>

Understanding this feature of tables in HTML is especially important when using tables as a design tool for your web page's layout. Coding this can be a little tricky at first, but will become much easier with a some experience.

The Cell Colors Attribute: BGCOLOR="#*nnnnnn*" (or "*color*")

The background color cells in a table can be specified with the attribute BGCOLOR="#nnnnnn" where *nnnnnn* represents the six-digit hexadecimal color code. Placing the attribute in the <TABLE> tag affects the entire table, placing the attribute in the <TR> tag affects the cells of the entire row, and placing the attribute in the <TD> tag affects that individual cell. The <TD> tag has the highest priority, overriding all other color settings, and the <TABLE> tag has the lowest priority. This holds true for any of the various attributes that these three tags have in common.

Other Table Tricks

There are several other aspects to tables in HTML that we can't cover in our class today. We encourage you to explore these by learning about them from a good HTML resource, whether it be online or in a reference book.

Practice Exercise

Go to the **Department Personnel** section of the *Bear Net-Works Department Information Sheet*. With the remaining class time, use the tags you have just learned to create a similar table for your Bear Net-Works web page.

Below is an example to get you started.

NAME	TITLE	PHONE	EMAIL
Jessica Rabbit	Dept. Directory	5-5555	jrabbit@ucmail
Ned Flanders	Clerk	5-1212	clerk@ucmail
William Gates	Tech	5-2000	tech@ucmail

Below is the code needed to display this example table. Try to create the table on your own first, and then go to the code afterwards if you need. When you've finished creating the basic table, use some of the other attributes we've learned to make it a bit more interesting.

```
<TABLE width="100%" border="1" cellpadding="3" cellspacing="2">
 <TR>
    <TD>NAME</TD>
    <TD>TITLE</TD>
    <TD>PHONE</TD>
    <TD>EMAIL</TD>
 </TR>
<TR>
    <TD>Jessica Rabbit</TD>
    <TD>Dept. Directory</TD>
    <TD>5-5555</TD>
    <TD>jrabbit@ucmail</TD>
 </TR>
<TR>
    <TD>Ned Flanders</TD>
    <TD>Clerk</TD>
    <TD>5-1212</TD>
    <TD>clerk@ucmail</TD>
 </TR>
<TR>
    <TD>William Gates</TD>
    <TD>Tech</TD>
    <TD>5-2000</TD>
    <TD>tech@ucmail</TD>
 </TR>
</TABLE>
```

Until Next Time...

We'll wrap up our introduction to HTML in our next class by learning a few more tags and attributes. You will each have time during the last part of that class to put into practice what you've learned.

At the end of this document, we've included an appendix that includes the code we've learned during this class, as well as some information on frames in HTML. Practice your HTML skills whenever you have a chance at work or at home.

We'll see you in *HTML 3*!

APPENDIX

Code Used in This Class

The <FONT> Tag and its Attributes

FACE=”font”	changes text font
SIZE=”n”	changes text size
COLOR=”n”	changes text color

The <TABLE> Tag and its Attributes

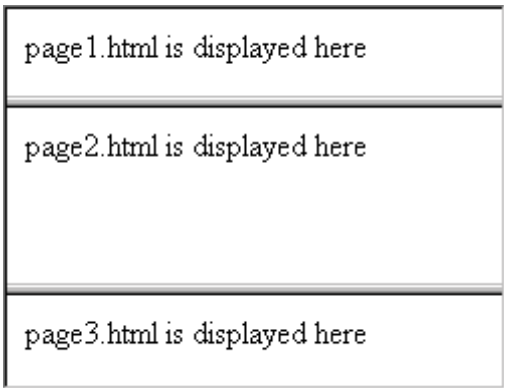
BORDER=”n”	specifies the thickness of a table border (if any)
CELLPADDING=”n”	specifies the amount of space between a cell’s contents and its borders
CELLSPACING=”n”	specifies the amount of space between cells
WIDTH=”n”	specifies the size of a table
HEIGHT=”n”	specifies the height of a table
ALIGN=”n”	aligns a cell’s contents horizontally
VALIGN=”n”	aligns a cell’s contents vertically
ROWSPAN=”n”	creates a cell that spans more than one row in a table
COLSPAN=”n”	creates a cell that spans more than one column in a table
BGCOLOR=”#nnnnnn”	changes the background color of a cell

Frames

FRAMES allow more than one HTML document to be displayed in a single browser window. In this new web series, we've opted not to teach Frames for several reasons–see the end of this appendix for a complete description. We've included the section on FRAMES from the original HTML 2 class here for those curious enough to tackle them on their own.

Creating Frames

The underlying concept of frames is used to divide the web browser window into independent sections, each section displaying its own file. Without frames, a user needs to switch back and forth between multiple files to view the information. The primary HTML file is the file referenced by the the web browser, but it does not directly appear on the webpage. Instead, it is used to set the divisions and specify which file is used for each section. In place of the <BODY> tags, the primary file uses the <FRAMESET> tags. Within the <FRAMESET> tags, <FRAME> tags are used to specify the files. Below is a simple frame code block with the resulting output displayed at right. Notice that each <FRAME> tag defines the HTML file to use.




```
<HEAD>
<TITLE>Frames Example</TITLE>
</HEAD>
<FRAMESET ROWS="25%,*,25%">
        <FRAME SRC="page1.html">
        <FRAME SRC="page2.html">
        <FRAME SRC="page3.html">
</FRAMESET>
```

Required <FRAMESET> Attributes

Defining the Frame Divisions - Rows and Columns

Within the FRAMESET tag, the ROWS=*n,n,...,n* and COL=*n,n,...,n* attributes are used to define the frame division of the web browser window where *n* can be pixel values, percentages, or a wildcard (*). Each value entered into the attribute defines a new frame space. Taking the example above, the browser window is divided into three separate rows. The first and last rows occupy 25% of the browser window.

When both the ROWS and COLS attributes are used, the browser window is divided into a grid like pattern. The wildcard specifies the middle row to occupy the remaining space. The following are a few more examples.

<FRAMESET ROWS="100,*">	The first row is 100 pixels, the second row occupies remaining space.
<FRAMESET COLS="50,100,*">	The first column is 50 pixels, the second column is 100 pixels, the third column occupies remaining space.
<FRAMESET ROWS="25%,50%,25%">	The first row occupies 25%, the second row occupies 50%, the third row occupied 25%. The is the same as the example discussed above, just a different format.
<FRAMESET COLs="100,*,10%">	The first column is 100 pixels, the last column is 10%, the second column occupies the remaining space.

By default, borders appear between each frame. A user can adjust the defined frame sizes by clicking and dragging the border. Scroll bars will appear whenever the frame content is too large for the frame.

The <FRAME> Tag

Within the <FRAMESET> tags, the <FRAME> tag is used to specify the HTML file. Just like images, the SRC attribute is used to reference the file. The order in which the each frame is specified begins at the top frame and moves left to right. Even if a frame is empty, it's recommended to at least have a file that specifies the background color to display.

```
<FRAMESET ROWS=30%,*,30% COLS="30%,*,30%">
        <FRAME SRC="page1.html">
        <FRAME SRC="page2.html">
        <FRAME SRC="page3.html">
        <FRAME SRC="page4.html">
        <FRAME SRC="page5.html">
        <FRAME SRC="page6.html">
        <FRAME SRC="page7.html">
        <FRAME SRC="page8.html">
        <FRAME SRC="page9.html">
</FRAMESET>
```

Optional <FRAME> Attributes

Resizing the Border Attribute: NORESIZE

The NORESIZE attribute sets the frame border so that it cannot be resized by the user. This attribute does not have any options.

*Controlling the Scroll Bars Attribute: SCROLLING="*yes|no|auto*"*

The SCROLLING attribute controls whether or not the scroll bar appears. The default option is auto, which causes the scroll bar to appear only when the frame content is too large for the frame space. When the option is set to ***yes***, a horizontal and vertical scroll bar will appear whether they are needed or not. With the option set to ***no***, scroll bars will never appear. If the content is too large, the viewer will not be able to see the material that is outside the frame space.

General Attributes

Changing the Border Size Attribute: FRAMEBORDER="yes|no"

The FRAMEBORDER attribute sets if the border should appear in the browser window. When the border does not appear, the viewer cannot resize the frame. Placing this attribute in the <FRAMESET> tag will affect the entire browser window. Placing the attribute in the <FRAME> tag will affect only the frame.

Changing the Border Color Attribute: BORDERCOLOR=#nnnnnn

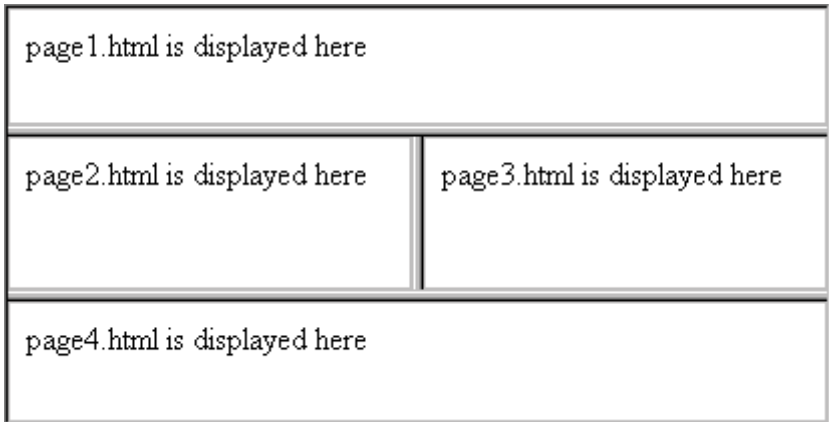
The border color of each frame can be specified with the attribute BORDERCOLOR=#*nnnnnn* where *n* represents the hexidecimal color code. Placing the attribute in the <FRAMESET> tag will affect the entire browser window. Placing the attribute in the <FRAME> tag will affect only the frame.

Changing the Frame Background Color Attribute: BGCOLOR="#nnnnnn"

The background color of each frame can be set with the attribute BGCOLOR= #*nnnnnn* where *n* represents the hexidecimal code. Placing the attribute in the <FRAMESET> tag will affect the embedded frames. Placing the attribute in the <FRAME> tag will affect only the frame.

Nested Framesets

The basic frameset tags using the columns and rows attribute divides the browser window into a grid pattern, similar to the standard table. When working with tables, it is possible to span rows and columns. Frames have the same feature, but it is accomplished by embedding a frameset container within other framesets. This is displayed in the following code block with the resulting output.

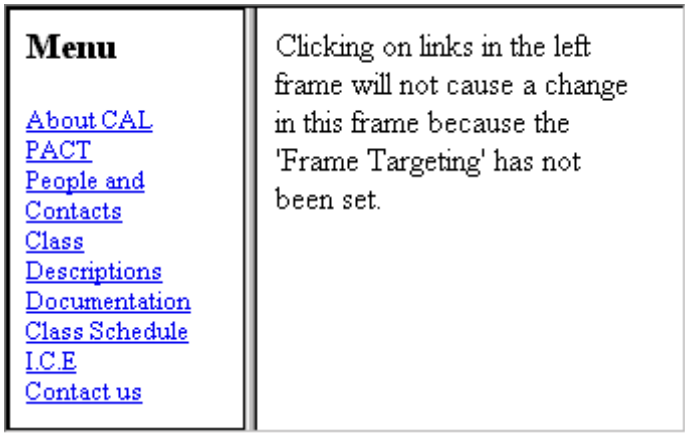



```
<FRAMESET ROWS="30%,*,30%">
  <FRAME SRC="page1.html">
     <FRAMESET COLS=50%,*>
          <FRAME SRC="page2.html">
          <FRAME SRC="page3.html">
     </FRAMESET>
  <FRAME SRC="page4.html">
</FRAMESET>
```

The first frameset divides the browser window into three rows. Remember that the <FRAME> tag begins at the top of the page. The second <FRAMESET> tag is used in place of the <FRAME> tag to divide the second row into the two columns. This process can be used to create any type of frame design desired.

Assigning Frame Names and Targetting Frames

When multiple frames are used within the browser window, the default action affects only the frame where the link resides. The example below displays two frames within the browser. The left frame contains all of the major links for the CAL PACT website. By default, if a viewer clicks on a link in the left frame, the new content will appear in the left frame, not in the right as shown below. It is not an appealing way to display information.



Changing the target for a link starts with assigning a name to each of the frames. This is done as an attribute for the <FRAME> tag in the main file which defines the frame. It is demonstrated in the following code:

```
<HEAD>
<TITLE>Frames Example</TITLE>
</HEAD>

<FRAMESET COLS="30%,*">
        <FRAME SRC="page1.html" NAME="nav">
        <FRAME SRC="page2.html" NAME="main">
</FRAMESET>
```

Here the left frame is assigned the name “nav” and right frame is assigned the name “main.” The links can now be targetted by using the frame name.

Note that frame names are case sensitive.

<BASE> Targetting *<BASE TARGET=”*name*”>*

The base tag controls the targetting for all links within a frame. Place the <BASE> tag directly after the <BODY> tag within your file. The format for the tag is <BASE TARGET=”*name*”> where *name* is the frame name to target. For the example, the following code block can be used for the HTML displayed in the left frame of the previous graphic.

```
<HTML>
<BODY>
<BASE TARGET="main">
<H3>Menu</H3>
<FONT SIZE=2>
        <a href="about.html">About CAL PACT</a><br>
        <a href="people.html">People and Contacts</a><br>
        <a href="descrip.html">Class Descriptions</a><br>
        <a href="document.html">Documentation</a><br>
        <a href="sched.html">Class Schedule</a><br>
        <a href="http://hrweb.berkeley.edu/ice/home">I.C.E</a><br>
        <a href="contact.html">Contact us</a><br>
</FONT>
</BODY>
```

*Individual Targetting <A HREF=”URL” TARGET=”*name*”>*

To define the target for a specific link, the TARGET attribute can be placed within the <A HREF> tag. Placing a target name within the <A HREF> tag will override any setting defined by a <BASE> tag.

Using the <NOFRAMES> Tag

After all this discussion about frames, it is good to remember that not all viewers have a browser that supports frames. If you choose to use the commands we have discussed so far, non-frames browsers will not be able to view your web page. This problem is solved by using the <NOFRAMES> tag. This tag is placed after the last </FRAMESET> tag. Within the <NOFRAMES> container, include the content that you would like the viewer will see.

Why is CAL PACT no longer teaching frames?

Reason #1

Because a frames page is comprised of three separate HTML documents, it is difficult to create a bookmark to a page deep within your site. For example, if the page contains a Navigation Bar in a separate frame, creating a bookmark to a page within this site will not bring up the Navigation Bar with that page. Instead, the link will only lead to a single HTML document. In order to load both the Navigation Bar and the page you wish to view into to the browser window, you need to link to the index.html file containing the <FRAMESET> tags. This file, however, loads the homepage with the Navigation Bar. What does this mean? This forces you to start from the homepage and search for the page you want in the site, *every single time*.

Reason #2

If the page you are viewing contains frames, resizing the browser window will sometimes cause the page you are viewing to be reloaded. This may cause them all to revert back to their initial values. In other words, it will return you to the homepage of the site, because the browser window will attempt to reload the index.html file. Why is this? When viewing a frames page, the browser is tricked into thinking it has only loaded one – the index.html file. This file, however contains the instructions that allow multiple frames to be displayed. Regardless of the page you view with in the site, so long as it contains more than one frame, the browser still believes it is displaying the index.html file. In this way, causing the browser to reload the index.html file will force the frames to display their default values (pages).

Reason #3

The various search engines on the Web have several ways of indexing new web sites. One way is to allow people to submit them via e-mail or a web form. Some search engines have engineered small programs that travel across the internet looking for new pieces of information to index and link to. The internet is no small place, so the automated approach may take a while, but within a few months, most new web sites have been indexed by the larger and more popular search engines.

These automated information finders are not terribly bright, however. Remembering Reason #1, we would want to make sure a search engine linked to our index.html file – to insure that our Navigation bar would get loaded with the rest of our site. This may not happen. Automated searching programs index and create links to content. In a frames page, the index.html file does not contain the content, only instructions for the browser on how to display its content (in this case it instructs the browser to display more than one frame). Because of this, the index.html file would not be linked to and a link to our site from a search engine would, therefore, not bring up our Navigation Bar.