



Web Design 2

Using Links, Tables, and more!

Web Design Series - XHTML part B

CAL People and Computer Training
University of California, Berkeley

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



Use this
space for notes

Introduction

Welcome to **XHTML part B**! If you're here today, you've already completed *Web Design 1*, which means you're ready to expand your XHTML skills. Today we'll continue to add to our XHTML skill set.

Skills you need for this course

- 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 *Web Design 1*

Skills and concepts you will learn in this class

- Links - Anchor, Image, Email
- Commenting in your code
- Skip Navigation
- Tables - the basics
- Accessibility Recommendations

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 **XHTML part B**, things will be a bit different 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 *XHTML 1* we learned how to create a webpage using a few basic XHTML tags. Today, we'll learn to create tables to list tabular data, different types of links, add comments inside your code, skip navigation and learn some tips for web page accessibility.

Your instructor should now direct you to our XHTML document template (it's our *index.html* file). It contains the code you worked with in XHTML 1. Open the file in a text editor and also open it using a web browser.

Links

Creating a Hypertext Link Using an Image

As you remember, a plain hypertext link appears in your code as follows:

```
<a href="http://www.berkeley.edu">TEXT GOES HERE</a>
```

To create a link using an image, we need to replace the link text (i.e., "TEXT GOES HERE") with an `` tag. Here's an example of the code:

```
<a href="http://www.berkeley.edu"></a>
```

The image will now be displayed, with a colored border, indicating it is a link. (The boldface in the code example above is for use in the highlighting only.)

Linking to an E-Mail Address

Linking to an e-mail address is very similar to a standard hypertext link. In place of an web address, however, we use an e-mail address in conjunction with the `mailto:` command. For example:

```
<a href="mailto:username@domain name">TEXT GOES HERE</a>
```

When displayed in a web browser, the link above will look identical to a hypertext link. Clicking on it will automatically start a visitor's pre-defined e-mail program, placing the identified e-mail address in the "TO:" field. However, the visitor's browser must be appropriately configured for this to work. If it is not properly configured (for instance, their browser has not been linked to their e-mail application), they will receive an error message. You may want to replace the "TEXT GOES HERE" with the actual email address for your site visitors.

Anchors: Linking to a Specific Point in a Web Page

A link as we have learned up to this point will simply take a viewer to the top of another web page. However, you may also create a link to a specific point within a page. You use an anchor to accomplish this. The first step is to find the point in the document where you want to take the person with the link. Next, a keyword is placed at that point as a reference for the link. Then it's time to create the link itself.

Imagine you would like to add the entire Class schedule, for both Fall and Spring, at the bottom of our Bear Net-Works homepage. To spare visitors the trouble of scrolling down to find the Spring Class Schedule each time they visit that page, we'll create a link at the top the homepage leading directly to it. To do this, we use the `<a>` tag in conjunction with the NAME attribute.

For example, at the line just above our Spring Class Schedule, we'll add the following code:

```
<a name="keyword"><strong>Spring Courses</strong></a>
```

The content you place between the opening and closing `<a>` tag will become the anchor point. The keyword you specify will be used to reference it. The text "keyword" in the example above can be any word you choose. However, if you create more than one anchor, you will need to use different keywords for each.

Next, to create the link to our anchor point, decide where on the page you would like the link to be placed (somewhere at the top in this case), and type the following code:

```
<a href="#keyword">LINK TEXT GOES HERE</a>
```

Important Note: Your keywords must match in order for the link to work.

Clicking on your link should now take you directly to the Spring Courses.

Linking to an Anchor in Another Document (Page) in our Site

If our Spring Courses were listed in a separate document called `classes.html`, we would create an anchor in that web page document in the same manner as before. When linking to that anchor, however, we would need to specify the name of the XHTML document the anchor was located in, as well as the anchor. For example:

```
<a href="classes.html#keyword">LINK TEXT GOES HERE</a>
```

Viewing HTML Source Code

While surfing the web, you may notice certain aspects of web pages you'd like to include in your own site. By selecting View|Source in your Internet Explorer browser window menu bar (View|Page Source in Netscape Navigator), you'll get to peek at the code used to create that page. It's a great way to learn new techniques and tricks, and to learn how others have written their HTML code.

Commenting Your Code

HTML allows you to include "hidden" text anywhere within a HTML document. This is called commenting. Comments might describe what you hoped to achieve by using a certain tag or to indicate what kind of content should be placed in a particular area. You might also use comments to show when and how to update, include, or remove specific content. If your site is a group project, you might use commenting to include editorial information.

While this text is hidden when the page is displayed in a web browser, people will still be able to see it if they choose to view the source code of your HTML document. Therefore, you should not include any private or sensitive information (passwords, etc.) in your comments.

To include comments, type your text between `<!--` and `-->`. This is a single tag, but may be as long as needed. Here's an example:

```
<!-- The following needs to be updated by August 23rd, ask Project manager for latest info update. -->
```

```
<ul>
```

```
    <li>Ned Flanders 510-488-2932</li>
```

```
    <li>Bart Simpson 510-382-9002</li>
```

```
</ul>
```

Skip Navigation

Skip navigation allows people using screen readers to quickly skip over repetitive navigation links to get to the main content of the page. Usually the main content in a web page is preceded by many links or other types of information. To have to tab through each link before getting to the main content is a very tedious process for those with screen readers. Some pages like Espn.com might require 30+ presses of the tab key to get to the main content, if it weren't for skip navigation.

When should skip navigation be used? If there are five or more navigation links and/or content that comes before the main content of the page then skip navigation should be used.

Here is an example of what skip navigation would look like at the top of a page:

```
<!-- top of a page -->  
<a href="#main">Skip Navigation</a>
```

```
<!-- many navigation links -->  
<a id="main" name="main"></a>
```

```
<!-- beginning of main content -->
```

Do these tags look familiar? It's an anchor link! The anchor should have both an id attribute and name attribute, because depending upon the application, the anchor will be recognized by all types of browsers.

Hiding the skip link

You may want to hide the skip link from users for aesthetic purposes. There are several techniques to hide the skip link, but we only have time to cover one. You can use a blank image as a filler for the skip link.

```
<a href="main">  
</a>
```

The image used is spacer.gif which is a blank image and given a 1x1 pixel dimension, so that it is invisible. The alt text will be read once the user presses the tab key and "skip navigation" will be heard by a screen reader. This is a fairly easy way to hide the skip link from users.

Other ways of hiding the skip link involve using the same foreground and background colors on the link text. Hiding the skip link by positioning it off screen using stylesheets. Visit <http://jimthatcher.com/skipnav.htm> for more information on how to create skip navigation.

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 learned that they could use them to control the layout of their pages. Using tables for laying out web pages became problematic as applications other than desktop computers (pda's, screen readers, etc.) tried to access these web pages. For now, we will create tables in XHTML with the intent of styling them in stylesheets using Cascading Style Sheets (CSS) in Dreamweaver.

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

<pre> <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> </pre>	<pre> Row 1 Column 1 Row 1 Column 2 Row 2 C olumn 1 Row 2 Column 3 Row 3 Column 1 Row3 Column 2 </pre>
--	---

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?

<pre> <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> </pre>	<pre> Row 1 Column 1 Row 1 Column 2 Row 2 Column 1 Row 2 Column 3 Row 3 Column 1 Row 3 Column 2 </pre>
--	--

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"`

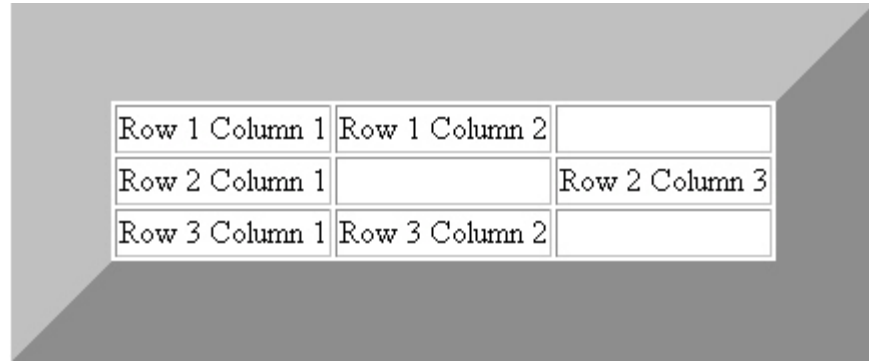
By default, the `<table>` tag creates a table with no border. While this is preferred when creating page layouts, sometimes a border is desired. To create a border, just add the **border** attribute to the `<table>` tag.

```

<table border="50">
...
</table>

```

For example, this code displays as follows on the next page.



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

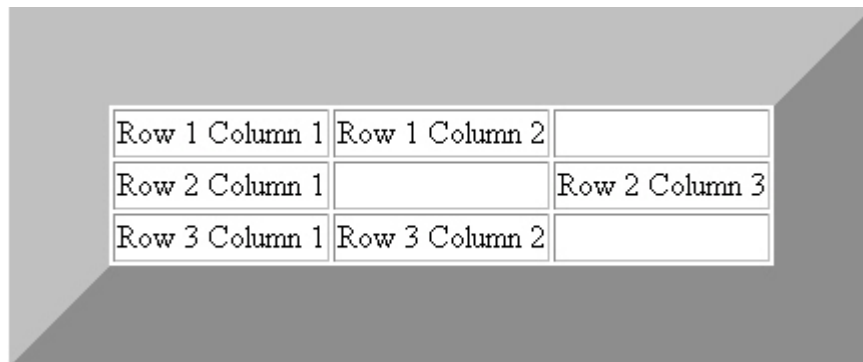
Notice the **border** attribute created an outer border of 50 pixels in size. This also creates borders inside the table. The cells that contain content are given a default border of two pixels, while the empty cells have no border at all. Sometimes, it may be aesthetically unpleasing to have borders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

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. if you want to do this, enter ** ** between the **<td>** tags.)

After adding this to our code, our new table should look like the one below.



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

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 of 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.

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.

Note



Alignment of tables should be done using CSS. This will be covered in a future Web Design class.

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. The **rowspan** attribute collapses one or more rows into one row. The **colspan** attribute collapses one or more columns into one column.

<td rowspan="2">...</td> This will collapse two rows into one.

<td colspan="3">...</td> This will collapse two columns into one.

It is hard to visualize until one gets to play around with it. Look at the example code on the next page to see how it is implemented.

```
<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 3</td>
  </tr>
</table>
```

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

Understanding this feature of tables in XHTML is especially important when using tables as a design tool for your web page's layout. Using tables solely as a design tool isn't recommended, but having this in your arsenal can be useful in many situations. Coding this can be a little tricky at first, but will become much easier with a some experience.

Table Headers

Most data tables have headers in the first row or column indicating the type of data being used. Whether it's data of dates, money or stats, a string of cells is usually organized under a proper header. Using the `<th>` tag allows you to create headers for your data tables.

To make table headers in our current table, replace the `<td>` tags with `<th>` tags in the first row. Doing this will make the content inbetween the header tags bold and centered. Screen readers will automatically treat `<th>` tags as proper headers and note it to the audience by emphasizing the text. This is why it is important to use header tags whenever possible, so that content is readily organized and accessible. Header tags can also be easily styled using style sheets in CSS. You are not stuck with only bold and centered header tags, but have many different options to change how they are presented.

Look below to see how header tags are used in a table.

```
<table border="1">
  <tr>
    <th rowspan="2">Row 1 and 2 Column 1</th>
    <th>Row 1 Column 2</th>
    <th>Row 1 Column 3</th>
  </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 3</td>
  </tr>
</table>
```

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

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="500" border="1">
  <tr>
    <th>NAME</th>
    <th>TITLE</th>
    <th>PHONE</th>
    <th>EMAIL</th>
  </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>
```

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.

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?

Web Design Course - Accessibility Recommendations

Here are some recommendations to keep your website accessible to the widest possible audience. Most of these will be included in the UC Berkeley Accessibility Policy currently being written for all UC Berkeley websites. The sooner you begin to adhere to these guidelines the easier it will become to code web pages that are lean and accessible to a wide audience.

Structure the content on a piece of paper before you begin coding.

- Map out your content and organize to reduce the number of clicks a user has to click in order to get to information.
- Navigation links should be simple and informative.
- Try to keep pages under five links deep, three is even better.

All non-text content should provide text equivalents.

- Provide a description of all images, sound files, applets, etc. in an **alt** text tag.
- If your page cannot contain text equivalents within it, consider creating a parallel text-only version of the page as a last resort.

Make sure that text and graphics make sense without color.

- People who cannot differentiate between colors or who are using devices with noncolor or nonvisual displays will not receive or understand information that relies on color for its meaning.
- Choose backgrounds that contrast with page text and don't interfere with readability of content.

Don't use frames.

- Frames are an out-dated form of creating layouts
- Frames are not universally accessible.
- The content of frames may not be searchable by search engines.

Use descriptive links.

- Instead of denoting a link with the words "Click here" or similar phrase, be descriptive when providing links; for example: "further information from the W3C on Web accessibility." (underlined words are a link)
- Consider allowing such links to stand on their own line or provide an ordered or unordered list of links in HTML.

Use headings and lists to organize your page and reinforce consistent page structure.

- When possible, use mark-up language (<h1>,e.g.) rather than images or visual cues to emphasize the structure of your page.
- Organize documents so they may be read without style sheets.

Test pages in multiple browsers, operating systems, and connection speeds.

- Test your pages on older Windows and Macintosh systems in both Netscape and Internet Explorer.
- Pages intended for modem users should be tested for load times on slower connections (56K). A large image that downloads immediately on University computers may take 30 seconds to load on a modem connection.

Provide alternate routes to information when using interactive technologies.

- Provide captioning and transcripts of audio and descriptions of video.
- Pages that use Flash, Shockwave, or other interactive elements may not be accessible with older browsers. Provide a link to a text-only or non-interactive version of the information. Use special effects with caution and only with good reason.

Provide Skip Navigation so users can tab quickly to the main content.

- Have “Next Page,” “Previous Page,” and “Up A Level” navigational meta tags
- Have “Skip to Main Content” links
- Enable keyboard shortcuts or hotkeys for link selection

Potential difficulties for web pages include:

- May not work (or will work unpredictably) on different systems
- Harder to implement and maintain
- Takes a long time to download
- Requires users to install plug-ins

Strongly Discouraged:

- Use of red/green combinations
- Designation of headings by using <bold> formatting
- Use of absolute font sizes in running text, e.g. “10pt.”
- Making navigation or other essential information depend solely on images

Until Next Time...

In the next Web Design 3 class we'll jump right into learning Dreamweaver MX 2004. So far, we've learned just enough XHTML to create content for our web pages. The next step is to add some style to them using style sheets (CSS) within Dreamweaver.

We've included an appendix that includes the code we've learned during this class. Practice your XHTML skills whenever you have a chance at work or at home.

We'll see you in Web Design 3!

APPENDIX

Code Used in This Class

The <TABLE> Tag and its Attributes

<code>border="n"</code>	specifies the thickness of a table border (if any)
<code>width="n"</code>	specifies the size of a table
<code>height="n"</code>	specifies the height of a table
<code>rowspan="n"</code>	creates a cell that spans more than one row in a table
<code>colspan="n"</code>	creates a cell that spans more than one column in a table
<code><th> </th></code>	creates a header in the table

Links

<code></code>	an image as an link
<code></code>	an email link
<code> </code>	
<code> </code>	an anchor link

Commenting your code

<code><!-- ... --></code>	adding comments within your code
---------------------------------	----------------------------------

Skip Navigation

<code>Skip Navigation</code>	skip navigation links
<code></code>	