**Web Site Development**

**Week 2 Content:**

**CHAPTER 9- Basic Tables……………………………………………………………………………...….………………..2**

9.1- Inserting a Table

9.2- Table Borders

9.3- Table Headers

Basic Tables- Actions

Basic Table Exercises

**CHAPTER 10- Iframes…………………………………….…………………………………………………………………6**

10.1- What is an Iframe?

10.2- Inserting Iframes

10.3- Setting Height and Width

10.4- Using an Iframe for a Link Target

Iframes- Actions

Iframe Exercises

**CHAPTER 11-forms……………………………………………………….……………………..…………………………….9**

**CHAPTER 12 -Video and Audio ……………………………….…………………………………………………..……….22**

**CHAPTER 13 -Troubleshooting ……………………………………………...…………………………………………….28**

**Chapter 9: Basic Tables**

**9.1- Creating a Table:**

A table serves as a structured container for presenting data in an organized manner. Conceptually, a table can be visualized as a rectangular shape bordered by four sides. Inside the table, there are intersecting columns (vertical) and rows (horizontal) that create individual cells. Each cell has its own four borders and can store various types of content, such as text and images. It is important to note that you can make table and cell borders either visible or invisible. It is no longer recommended to use tables for defining webpage layouts; instead, they should be used exclusively for displaying data. In HTML, tables are composed of the following basic components: `<table>…</table>` (defines the table itself), `<tr>…</tr>` (defines a table row), and `<td>…</td>` (defines table data or cell content).

Before you start adding code to create your table, it's helpful to have a clear idea of how you want your final table to appear. Consider sketching a rough layout on paper, indicating where you intend to place various elements and data. This preliminary planning will save you time when you begin coding.

To insert a table into your webpage, start by placing the `<table>` start tag where you want the table to appear. Next, define your table's rows and the data within the cells. It's a good practice to keep row and data tags on separate lines to improve code readability and facilitate future modifications. Add a row by using the `<tr>` start tag, followed by creating individual cells using the `<td>…</td>` tags to enclose the data for each cell. Complete the row by inserting the `</tr>` end tag. Repeat this process to add more rows. When you have finished defining your table, close it with the `</table>` end tag.

Table Attributes, Descriptions, and Examples:

A screenshot of a computer screen

Description automatically generated

Example:

<table>

<tr>

<td>Red</td>

<td>Black</td>

<td>Blue</td>

</tr>

<tr>

<td>Green</td>

<td>Gold</td>

<td>Silver</td>

</tr>

</table>

Result:

Red Black Blue

Green Gold Silver

**9.2- Table Borders:**

By default, tables you create do not have visible borders, meaning there is no visible line defining the table's boundaries. Adding borders can enhance the appearance of your table and make it visually distinct. To assign a border to your table, include the BORDER attribute within the `<table>` start tag and specify the desired border thickness in pixels. The specified thickness will apply to the table border but not to individual cell borders. If you want to change the border color, you can also use the BORDERCOLOR attribute within the `<table>` start tag and set the desired color value. This color will be applied to both the table border and individual cell borders. While you can style your table within your HTML code, it's generally advisable to manage most of the styling using cascading style sheets (CSS), which will be covered in a later chapter.

Table Attributes, Descriptions, and Examples:

A screenshot of a computer program

Description automatically generated

Example:

<table border="2" bordercolor="#000000">

<tr>

<td>Red</td>

<td>Black</td>

<td>Blue</td>

</tr>

<tr>

<td>Green</td>

<td>Gold</td>

<td>Silver</td>

</tr>

</table>

Result:

```

Red Black Blue

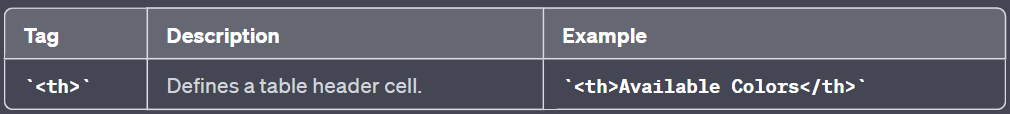
Green Gold Silver

```

**9.3- Table Headers:**

You can include a header at the top of your table to identify or label the table's contents. Table headers typically appear in a dedicated cell at the beginning of the table, with the text being both bold and centered. To add a table header, use the `<th>…</th>` tags. Place these tags after, but not within, the `<table>` start tag.

Table Attributes, Descriptions, and Examples:



Example:

<table border="2" bordercolor="#000000">

<th>Available Colors</th>

<tr>

<td>Red</td>

<td>Black</td>

<td>Blue</td>

</tr>

<tr>

<td>Green</td>

<td>Gold</td>

<td>Silver</td>

</tr>

</table>

Result:

```

Available Colors

Red Black Blue

Green Gold Silver

**Chapter 10: Iframes**

**10.1 - Understanding Iframes:**

An Inline Frame (Iframe) is an independent HTML document that you embed within your main HTML document. Typically, Iframes are used to display content from other webpages, such as videos or advertisements. Iframes can also be customized to act as a miniature webpage within your own, complete with its scrollbar that doesn't affect the main page.

**10.2 - Inserting an Iframe:**

To insert an Iframe into your webpage, you need to use both the opening and closing tags `<iframe>...</iframe>` and specify the source URL using the `src` attribute, much like inserting an image.

Attributes Table:

A screenshot of a computer code

Description automatically generated

**10.3 - Setting Height and Width:**

You can define the dimensions of your Iframe by including the `height` and `width` attributes within the opening tag. These dimensions can be in pixels or as a percentage of the browser window width.

Attributes Table:

A screenshot of a computer program

Description automatically generated

**10.4 - Using an Iframe for a Link Target:**

You can set a link within your page to open within an Iframe by using the `TARGET` attribute. To do this, you must name your Iframe using the `NAME` attribute. The values of `target` and `name` must match.

Attributes Table:

A screenshot of a computer code

Description automatically generated

Actions Summary:

Here are the summarized actions for working with Iframes:

Inserting an Iframe:

1. Place `<iframe src="?"></iframe>` where you want the Iframe.

Setting Height and Width:

1. Inside the `<iframe>` tag, add `height="X"` and `width="Y"` attributes.

Using an Iframe for a Link Target:

1. In the `<iframe>` tag, specify `name="??"`.

2. Add `<a href="??" target="iframe\_name">Link Name</a>` where you want the link.

**Chapter 11 Forms**

**11.1- Introduction to Forms:**

Forms are used to collect information from people who visit your website. For example, you can use forms to find out details about your visitors through surveys and feedback or engage in e-commerce by selling your goods and services to people.

Forms are defined by the `<form>…</form>` tags and are made up of different elements to collect data, such as text boxes and radio buttons. Once the user inputs all of the information, they submit the form, using the “submit” button that you create. What happens with the data is a decision you will need to make. You can use a script to manage the data, send the data to a database, or even receive the data via email.

Most forms are processed using CGI scripts, CGI (Common Gateway Interface) is a script written in a language such as Java or Perl and runs on a Web server. Most Web servers accommodate the processing of CGI scripts, but you should check first with your Web host to make sure. In addition, you will want to find out the location of the server’s CGI-bin (a directory where CGI scripting is stored), as this is where you will need to store the CGI script you create or use. If you know a language such as Perl, you can write your own script. There are also hundreds of free scripts available online that you can use, such as the ones at sites like The CGI Resource Index (http://www.cgi.resourceindex.com) and JavaScript Kit (http://www.javascriptkit.com). You will need to make the necessary changes to the CGI script that you use (script variables, path names, etc.) and upload the CGI script to your Web host server.

To begin creating a form using a CGI script, start with the `<form>` tag, containing the command `method="post"` and the `action` attribute, with a value equal to the path and name of your CGI script.

Start Tag: `<form method="post" action="?">`

End Tag: `</form>`

Attributes:

- `method="post"`

- `action="?"` (Where "?" is the path and name of your CGI script for the form.)

Example:

```html

<form method="post" action="/cgi-bin/contact.pl">

</form>

**11.2- Sending Form Data via Email:**

If your Web server does not support CGI scripting, or if you prefer to avoid it altogether, you can send the form data directly to an email address. Keep in mind that this is not a good solution if you are capturing sensitive data such as credit card numbers, as it is not a secure form of transmission. Sending your form data to an email address is a good solution if your form is simple.

To send the information to an email address, you use the following coding:

Start Tag: `<form method="post" enctype="text/plain" action="mailto: ?">`

End Tag: `</form>`

Attributes:

- `method="post"`

- `enctype="text/plain"`

- `action="mailto: ?"` (Where "?" is the email address you wish to send the form to.)

Example:

<form method="post" enctype="text/plain" action="mailto:info@yourcompany.com">

</form>

**11.3- Text Boxes:**

Text boxes are the most basic elements that forms use in the collection of data. Text boxes are typically used when the input requires a single line of text. To create a text box, you use the `<input>` element tag and `type="text"` attribute and place it between the `<form>…</form>` tags. You must also specify a unique name for the text box using the `name` attribute.

By default, text boxes are 20 characters wide. You can change the width of the field that is displayed by using the `size` attribute. You can also limit the number of characters the user can type into the text box by using the `maxlength` attribute.

Start Tag: `<input>`

No End Tag.

Attributes:

- `type="text"` (Identifies the element as a text box.)

- `name=` (Required.)

- `size=`

- `maxlength=`

Example:

<form method="post" action="cgi-bin/contact.pl">

<br>Your Full Name:

<input type="text" name="fullname" size="50" maxlength="45">

</form>

Result:

Creates a text box called “fullname” that is 50 characters in length and holds a maximum of 45 characters of input.

**11.4- Text Areas:**

Sometimes, you will want to collect text from individuals that requires a larger box. This is common in situations where you ask for feedback that may require multiple sentences. In this case, you can insert a large text area using the `<textarea>…</textarea>` element tags and assigning a unique (and required) name using the `name` attribute.

You control the dimensions of the text area using the `rows` and `cols` attributes, measured in relation to the character height. You can also control how text wraps within the text area using the `wrap` attribute. You have three choices when assigning a value to the `wrap` attribute: “soft,” “hard,” and “off.” A value of “soft” will wrap the text in your area but will not wrap text in the form results (meaning it will be in a single field in a database or a single line in an email). The value “hard” wraps text in both the text area and form results. The value “off” turns text wrapping completely off and forces users to create new lines using the “Enter” key on their keyboard.

If the user types more characters than can be seen in the text area you created, scroll bars will appear to enable viewing of the text. Text areas hold up to 32,700 characters.

Start Tag: `<textarea>`

End Tag: `</textarea>`

Attributes:

- `name=` (Required.)

- `rows=` and `cols=`

- `wrap=` (“soft,” “hard,” or “off”)

Example:

<form method="post" action="/cgi-bin/contact.pl">

<br>Your Comments:

<textarea name="comments" rows="15" cols="75" wrap="hard"></textarea>

</form>

Result:

Creates a text area called “comments” that is 15 rows high and 75 columns wide and forces a “hard” wrap, where text in the text area and form results are wrapped.

**11.5- Check Boxes:**

Check boxes are used when you want visitors to select from one or more options that you present. To create check boxes, you use the `<input>` element tag and `type="checkbox"` attribute and place it between the `<form>…</form>` tags. You can group the check boxes using the same `name` attribute for each element. Remember that the value you assign is a description of the checkbox. If you want your check boxes to all appear on a separate line, use the `<p>` or `<br>` tags before each element.

Start Tag: `<input>`

No End Tag.

Attributes:

- `type="checkbox"` (Identifies the checkbox.)

- `name=` (Required.)

- `value=`

Example:

<form method="post" action="/cgi-bin

/contact.pl">

<br>Have you purchased from us before?

<br><input type="checkbox" name="purchase" value="yes">Yes

<br><input type="checkbox" name="purchase" value="no">No

</form>

Result:

Have you purchased from us before?

Yes

No

**11.6- Menu Lists:**

Menu lists are typically used when you have a long list of choices to give users. The menu list appears as a drop-down list and allows the user to select their choice. Between the `<form>…</form>` tags, use the `<select>` element tag, along with the `name` attribute and define a size for the box with the `size` attribute (measured in character lines). The `<option>` element tag is then used to define each of the choices in the menu.

Start Tag: `<select>`

End Tag: `</select>`

Attributes:

- `name=` (Required.)

- `size=` (Measured in character lines.)

Supporting Start Tag: `<option>`

No End Tag.

Attributes: `value=` (Name of selection.)

Example:

<form method="post" action="/cgi-bin/contact.pl">

<br>Which best describes your status?

<select name="status" size="1">

<option value="Business">Business

<option value="Government">Government

<option value="Individual">Individual

</select>

</form>

Result:

Which best describes your status?

- Business

- Government

- Individual

**11.7- Radio Buttons:**

Radio buttons are the small circles (O) that appear in forms. Radio buttons allow you to present a series of choices (grouped under the same `name` attribute) but only allow the user to select a single choice. To add radio buttons, use the `<input>` element tag between the `<form>…</form>` tags with a `type` value of “radio.” The `<p>` or `<br>` tags allow you to position each choice on its own line.

Start Tag: `<input>`

No End Tag.

Attributes:

- `type="radio"`

- `name=` (Required.)

- `value=`

Example:

<form method="post" action="/cgi-bin/contact.pl">

<br>How did you hear about us?

<br><input type="radio" name="source" value="tvradio">TV, Radio

<br><input type="radio" name="source" value="print">Print Ad

<br><input type="radio" name="source" value="internet">Internet Search

</form>

Result:

How did you hear about us?

- TV, Radio

- Print Ad

- Internet Search

**11.8- Submit Button:**

The submit button is a required piece of your form since when it is clicked, it actually sends the data to be processed. You must have a submit button in your form in order to receive the data. Using the `<input>` element tag and a value of “submit” for the `type` attribute, assign a `value` that is the text you want to appear in the button (typically “Submit”).

Start Tag: `<input>`

No End Tag.

Attributes:

- `type="submit"`

- `value=` (Text in button.)

Example:

<form method="post" action="/cgi-bin/contact.pl">

<!-- Form elements go here -->

<br><input type="submit" value="Submit">

</form>

Result:

Submit

**11.9- Reset Button:**

The reset button is used to allow a user to clear all of the information they have entered into a form. To insert a reset button, use the `<input>` element tag with a `type` attribute and a value of “reset,” and the `value` attribute with a value that is the text you want to appear in the button (typically “Reset”).

Start Tag: `<input>`

No End Tag.

Attributes:

- `type="reset"`

- `value=` (Text in button.)

Example:

<form method="post" action="/cgi-bin/contact.pl">

<!-- Form elements go here -->

<br><input type="reset" value="Reset">

</form>

Result:

Reset

Clears all form data.

**11.10- Changing the Tab Order:**

Users can navigate through the elements of your form by using the Tab key on their keyboard, jumping from one element to the next. By default, the order the Tab key follows is the order in which you entered the form elements in your HTML page. To change the tab order, add the `tabindex` attribute to your element tags and assign a numeric value for the position you want (“1” for 1st, “2” for 2nd and so on.)

Start Tag: Any element tag in your form.

Attributes: `tabindex=`

Example:

<form method="post" action="/cgi-bin/contact.pl">

<br>How did you hear about us?

<br><input type="radio" name="source" value="tvradio" tabindex="1">TV, Radio

<br><input type="radio" name="source" value="print" tabindex="2">Print Ad

<br><input type="radio" name="source" value="internet" tabindex="3">Internet Search

</form>

Result:

How did you hear about us?

- TV, Radio

- Print Ad

- Internet Search

Tab order follows your specifications.

**Actions - Forms**

**Create a Form That Utilizes a CGI Script:**

1. To incorporate a form, position your cursor where you want it in the document, and commence the form with the opening tag: `<form`.

2. Insert a space and specify the attributes: `method="post"` and `action="?"` (replace "?" with the path and name of your CGI script for the form).

3. Complete the tag by adding: `>`.

4. On the subsequent line, conclude the form by typing: `</form>`.

**Create a Form That Directly Sends Information to an Email Address:**

1. To embed a form in your content, begin by typing the initial portion of the opening tag: `<form`.

2. Add a space and include the attributes: `method="post"`, `enctype="text/plain"`, and `action="mailto:?"` (replace "?" with the recipient's email address to receive the form data).

3. Seal the tag by typing: `>`.

4. Proceed to the next line and finalize the form with: `</form>`.

**Add a Text Box to a Form:**

1. Inside the `<form>…</form>` section, introduce a text box by typing: `<input type="text" name="?"` (replace "?" with a unique name for the text box).

**Add a Text Area to a Form:**

1. In the desired position within the form, start with the initial segment of the opening tag: `<textarea name="?"` (substitute "?" with the required unique name for the text area).

2. Conclude the tag with: `>`.

3. On the subsequent line, conclude the text area with: `</textarea>`.

**Incorporate Check Boxes in a Form:**

1. At the point where you want to insert a check box in the form, input: `<input type="checkbox" name="X">Y` (replace "X" with the necessary name for the checkbox, and "Y" with the text to be displayed alongside the checkbox).

**Include a Menu List in a Form:**

1. In the location within the form where you intend to add a menu list, commence with the opening tag: `<select name="?" size="X">` (replace "?" with the mandatory unique name for the menu list and "X" with the size of your menu list box in character lines).

2. On the following line, identify the initial choice by entering the element tag: `<option value="X">Y` (replace "X" with the option's value and "Y" with the text representing the choice).

3. Repeat steps 1 and 2 for each additional choice.

4. On the last line, conclude the menu list with: `</select>`.

**Integrate Radio Buttons into a Form:**

1. To introduce radio buttons into the form, input: `<input type="radio" name="X">Y` (replace "X" with the obligatory name for the radio button and "Y" with the accompanying text).

**Add a Required Submit Button to a Form:**

1. In the section of the form where you wish to include the submit button, type: `<input type="submit" value="?">` (replace "?" with the desired text to appear on the submit button).

**Incorporate a Reset Button into a Form:**

1. At the location within the form where you want to insert the reset button, input: `<input type="reset" value="?">` (replace "?" with the text you want to display on the reset button).

**Modify the Tab Order:**

1. Within each element tag, add the attribute: `tabindex="?`" (replace "?" with the numerical value corresponding to the element's position in the tab order).

CHAPTER 12-

Video and Audio

**12.1 - Incorporating Video and Audio Files:**

Tag Attributes for <video> and <audio> Elements:

- `<video>` and `<audio>` elements can have the following attributes:

- `height`: Sets the height of the video or audio element.

- `width`: Sets the width of the video or audio element.

- `controls`: Adds video/audio controls (play, pause, volume, etc.).

Example 1 - Embedding Video Using <video> Element:

<video width="480" height="360" controls>

<source src="video.mp4" type="video/mp4">

<source src="video.webm" type="video/webm">

Your browser does not support the video tag.

</video>

Note: “Your browser does not support the video “ This line is placed within the <video> element, but outside of the <source> elements. It's a fallback message that will be displayed if the browser does not support the HTML5 video tag or if it can't play any of the specified video formats.

Example 2 - Embedding Audio Using <audio> Element:

<audio controls>

<source src="audio.mp3" type="audio/mpeg">

<source src="audio.ogg" type="audio/ogg">

Your browser does not support the audio tag.

</audio>

**12.2 - Linking to Video and Audio Files:**

Tag Attributes for `<a>` Element:

- `<a>` elements can have the following attribute:

- `href`: Defines the hyperlink reference or link target.

Example 3 - Creating a Link to an Audio File:

<a href="audio.mp3">Listen to our audio</a>

The simplest way to provide audio and video content to your website visitors is by creating direct links to the media files. When users click these links, the files will open and play in a separate window, just like regular web links. To avoid broken links, ensure that you've uploaded the video or audio files to your web server.

**12.3 - Adding Video:**

Tag Attributes for <video> Element:

- `<video>` elements can have the following attributes:

- `height`: Sets the height of the video element.

- `width`: Sets the width of the video element.

- `controls`: Adds video controls (play, pause, volume, etc.).

- `src`: Defines the link to the video file.

- `type`: Tells the browser what type of file you are using.

Tag Attributes for <source> Element:

- `<source>` elements can have the following attributes:

- `src`: Defines the link to the video file.

- `type`: Tells the browser what type of file you are using.

Example 4 - Embedding Video with <video> Element:

<video width="480" height="360" controls>

<source src="video.mp4" type="video/mp4">

<source src="video.webm" type="video/webm">

Your browser does not support the video tag.

</video>

When incorporating video into your webpages, you need to follow several steps. Along with the `<video>` element, you must include a `<source>` element that specifies the video file (SRC attribute) and its type (TYPE attribute). Currently, the major video formats supported are MP4, WebM, and Ogg. To enhance compatibility, consider including multiple file types in your HTML code.

It's crucial to include HEIGHT and WIDTH attributes in the `<video>` start tag to reserve space for the video during page loading. Without these attributes, the page layout may shift while the video loads. The CONTROL attribute is used to add standard video controls like play, pause, and volume.

To enhance the chances of video playback on various browsers, you can combine the `<video>` element with the `<embed>` element. Additionally, it's advisable to include descriptive text between the `<video>` tags to inform users when the `<video>` element isn't supported by their browser.

**12.4 - Adding Audio:**

Tag Attributes for <audio> Element:

- `<audio>` elements can have the following attributes:

- `controls`: Adds audio controls (play, pause, volume, etc.).

- `src`: Defines the link to the audio file.

- `type`: Tells the browser what type of file you are using.

Tag Attributes for <source> Element:

- `<source>` elements can have the following attributes:

- `src`: Defines the link to the audio file.

- `type`: Tells the browser what type of file you are using.

**Example 5 - Embedding Audio with <audio> Element:**

<audio controls>

<source src="audio.mp3" type="audio/mpeg">

<source src="audio.ogg" type="audio/ogg">

Your browser does not support the audio tag.

</audio>

Incorporating audio into webpages follows a process similar to video. The `<audio>` element is used, and you should also include a `<source>` element with the SRC and TYPE attributes to specify the audio file and its format. Common audio formats are MP3, WAV, and Ogg. Including multiple file versions improves the likelihood of successful playback.

The CONTROL attribute adds basic audio controls like volume, play, and pause to your audio file. Just as with videos, you can combine the `<audio>` element with the `<embed>` element for broader compatibility. It's a good practice to provide descriptive text between the `<audio>` tags for users whose browsers don't support the audio format.

**Actions for Video and Audio:**

Linking to a Video or Audio File:

1. To create a link to a video or audio file, position the cursor where you want the link, and input the following HTML tag: `<a href="?">Y</a>`, where "?" represents the path and file name of the file, and "Y" is the text for your link.

Adding Video:

1. To insert a video into your content, place the cursor where you want the video to appear and initiate the opening tag: `<video`.

2. Specify the width and height of the video by including the attributes `width="X"` and `height="Y"`, where "X" and "Y" are values in pixels.

3. Enable video controls by adding the attribute `controls`.

4. To include the video source file, employ the `<source>` element with attributes `src="?`" and `type="video/mp4"` OR `type="video/webm"` OR `type="video/ogg"`, with "?" representing the path and file name of the video file.

5. For older browsers, insert an embedded video by using the `<embed>` element with `src="?`, where "?" is the path and file name of the file.

6. Add text to describe unsupported file types after the last line.

7. Conclude the video section with the closing tag: `</video>`.

Adding Audio:

1. To include audio in your content, position the cursor where you want the audio to be placed and start with the opening tag: `<audio`.

2. Enable audio controls by including the attribute `controls`.

3. Specify the audio source file by using the `<source>` element with attributes `src="?`" and `type="audio/mp3"` OR `type="audio/wav"` OR `type="audio/ogg"`, with "?" representing the path and file name of the audio file.

4. For older browsers, insert an embedded audio file using the `<embed>` element with attributes `src="?`, `width="X"`, and `height="Y"`, where "?" is the path and file name of the file, and "X" and "Y" are values in pixels.

5. Add descriptive text for unsupported file types after the last line.

6. Conclude the audio section with the closing tag: `</audio>`.

Using YouTube to Display Video:

1. To integrate a YouTube video into your content, choose the desired location, and select the code from the "Share" or "Embed" section of the YouTube video page.

2. Paste the copied code into your HTML document at the chosen location.

Create a table in HTML includes different types of elements like iframe tags showing embedded videos, map images, text links as shown in the following screenshot.

"Build an HTML table that incorporate various elements, such as iframe tags displaying embedded videos, map images, and text links, mirroring the elements depicted in the provided screenshot."

**Troubleshooting**

**13.1 - Addressing Issues:**

If you've ever encountered a malfunctioning website, you can empathize with the frustration users feel when they encounter problems. It's crucial to test your website thoroughly to prevent such issues. By investing time in meticulous review, you can steer clear of the common pitfalls, which include:

**1. Typo Errors:** Much like any programming language, HTML demands precision for correct functionality. A single mistyped or omitted character can wreak havoc on your webpages. Always engage in proofreading your documents and diligently verify the presence of closing tags and other essential elements.

2. **Broken Links**: When a user clicks a link that doesn't lead them to the intended destination, it's labeled a "broken link." During webpage testing, it's imperative to click on each link to ensure proper functionality. If an issue arises, verify that both the path and file name are accurate and that the linked file has indeed been uploaded to your web server.

3. **Missing Images:** The infamous red "x" that replaces an image is another common error. It's essential to confirm that the path and file name of the image are absolutely correct, encompassing the file extension (.jpg, .gif, etc.). Additionally, ensure that the image has been successfully uploaded to your web server.

If all else fails and you find it challenging to identify the problem, consider seeking assistance from another seasoned web developer. Sometimes, a fresh set of eyes can detect mistakes that may have escaped your notice. Furthermore, remember that, as discussed earlier in this manual, the internet offers a plethora of valuable resources. Analyzing HTML pages created by other developers can provide you with new ideas and inspiration for your own design ventures.