

A CONDENSED INTRODUCTION TO HTML/XHTML

(TEXT CHAPTER 2)

Origins and Evolution of HTML

- HTML was defined with SGML
- Original intent of HTML: for general layout control of documents that could be displayed by a wide variety of computers
- Recent versions:
 - HTML 4.0 – 1997
 - » Introduced many new features and deprecated many older features
 - HTML 4.01 - 1999 - A cleanup of 4.0
 - XHTML 1.0 - 2000
 - » 4.01 redefined using XML, instead of SGML
 - XHTML 1.1 – 2001
 - » Modularized 1.0, and drops frames

Origins and Evolution of HTML

(continued)

- Reasons to use XHTML vs. HTML:
 1. HTML has **lax syntax rules**, leading to *sloppy* and *ambiguous* documents
 - XHTML syntax is more strict, leading to *clean* and *clear* documents in a *standard* form
 2. HTML processors do not even enforce the few syntax rule that do exist in HTML
 3. The syntactic correctness of XHTML documents can be **validated**
- **However ... (don't take them yet!)**

The Opposite Side: HTML vs. XHTML

- Although the value of the consistent and coherent syntax rules of XHTML were widely recognized and accepted, but its **draconian error handling was not so accepted**, thus XHTML documents are served as HTML – a compromise.
- Therefore, in 2009, work on XHTML 2.0 was stopped; W3C took over development of HTML5
- In our textbook, HTML5 is readopted, but with XHTML syntax rules (which has advantage)
 - So in my lectures

Basic Syntax

- Elements (as containers of contents) are defined by tags (markers)
 - Tag format:
 - » Opening tag: `<name>`
 - » Closing tag: `</name>`
 - The opening tag and its closing tag together specify a container for the *content* they enclose

Basic Syntax (continued)

- Not all tags have content
 - If a tag has no content, its form is `<name />`
- The container and its content together are called an ***element***
- If a tag has **attributes**, they appear between its name and the right bracket of the opening tag
- Comment form: `<!-- ... -->`
- Browsers **ignore** comments, unrecognizable tags, line breaks, multiple spaces, and tabs
- Tags are only **suggestions** to the browser, even if they are recognized by the browser
- In XHTML, element & attribute names must be lowercase letters
- In HTML, can be any combination of uppercase & lowercase

Standard HTML Document Structure

- Every HTML document must begin with:

`<!DOCTYPE html>`

- `<html>`, `<head>`, `<title>`, and `<body>` are **required**
- The whole document must have `<html>` as the **root**
- `html` in an XHTML doc “**must**” have the `xmlns` attribute:

`<html xmlns = "http://www.w3.org/1999/xhtml">`

- A document consists of a **head** and a **body**
- The `<title>` tag is to give the document a title, normally displayed in the browser's window title bar (at the top of the display)
- The `meta` tag is used to provide the character set used

`<meta charset = "utf-8" />`

Basic Text Markup

- Text is normally placed in paragraph elements
- *Paragraph Elements*
 - The `<p>` tag breaks the current line and inserts a blank line - the new line gets the beginning of the content of the paragraph
 - The browser puts as many words of content as will fit in each line
- e.g.:

```
<!DOCTYPE html>
<!-- greet.html: A trivial document
-->
<html lang = "en">
  <head>
    <title> Our first document </title>
    <meta charset = "utf-8" />
  </head>
  <body>
    <p>
      Greetings from your Webmaster!
    </p>
  </body>
</html>
```


Basic Text Markup (continued)

- Line breaks
 - The *visual effect* of the `
` tag is same as that of `<p>`, except for the blank line
- Example of paragraphs and line breaks

On the plains of hesitation `<p>` bleach the
bones of countless millions `</p>
`
who, at the dawn of victory `
` sat down
to wait, and waiting, died.
- Typical display of this text:

On the plains of hesitation

bleach the bones of countless millions

who, at the dawn of victory
sat down to wait, and waiting, died.
- To Preserve whitespace, simply use `<pre>` elements

Basic Text Markup (continued)

- *Headings*
 - Six sizes, 1 - 6, specified with <h1> to <h6>
 - 1, 2, and 3 use font sizes that are larger than the default font size
 - 4 uses the **default size**
 - 5 and 6 use smaller font sizes

<!-- headings.html

An example to illustrate headings

-->

<html xmlns = "<http://www.w3.org/1999/xhtml>">

<head> <title> Headings </title>

</head>

<body>

<h1> Aidan's Airplanes (h1) </h1>

<h2> The best in used airplanes (h2) </h2>

<h3> "We've got them by the hangarful" (h3) </h3>

<h4> We're the guys to see for a good used airplane (h4) </h4>

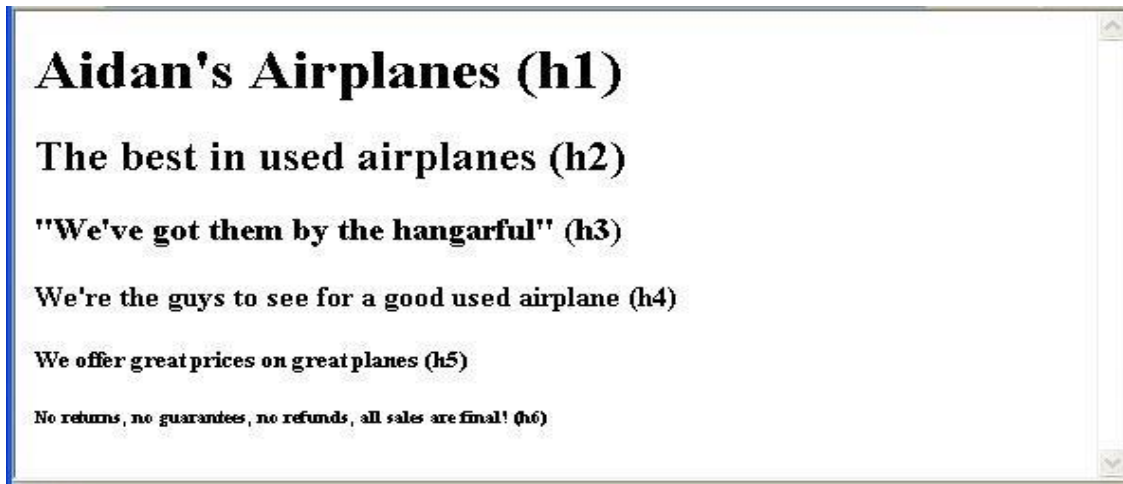
<h5> We offer great prices on great planes (h5) </h5>

<h6> No returns, no guarantees, no refunds, all sales are final (h6) </h6>

</body>

</html>

Basic Text Markup (continued)



- Blockquotes
 - Content of `<blockquote>`
 - To set a block of text off from the **normal flow** and appearance of text
 - Browsers often indent, and sometimes italicize
- *Font Styles and Sizes (can be nested)*
 - Boldface - ``
 - Italics - ``
 - Larger - `<big>`
 - Smaller - `<small>`
 - Monospace - `<tt>` or `<code>` (often displayed in Courier)
- *To preserve whitespace: `<pre>` `</pre>`, otherwise, ` `; ` `; ` `;*

Basic Text Markup (continued)

```
The <big> sleet <big> in <big> <i> Crete  
</i><br /> lies </big> completely </big>  
in </big> the street
```

The sleet in *Crete*

lies completely in the street

» These tags are not affected if they appear in the content of a `<blockquote>`, unless there is a conflict (e.g., italics)

– *Superscripts and subscripts*

» Subscripts with `<sub>`

» Superscripts with `<sup>`

Example: `x₂³`

Display: x_2^3

- Inline versus block elements (What are they?)

– Block elements **CANNOT** be nested in inline elements like `<h2>`

Basic Text Markup (continued)

- The above font size and font style stuff can be done with [style sheets](#), but these tags are not yet deprecated
- **Character Entities**

| <i>Char.</i> | <i>Entity</i> | <i>Meaning</i> |
|--------------|---------------|--------------------|
| & | & | Ampersand |
| < | < | Less than |
| > | > | Greater than |
| ” | " | Double quote |
| , | ' | Single quote |
| ¼ | ¼ | One quarter |
| ½ | ½ | One half |
| ¾ | ¾ | Three quarters |
| ° | ° | Degree |
| (space) | | Non-breaking space |
| @ | © | Copyright |
| € | € | Euro |

Basic Text Markup (continued)

- Horizontal rules: `<hr />` draws a line across the display, after a line break
- The `meta` element (for search engines) used to provide additional information about a document, with attributes, but no content!
- E.g.,

```
<head>
```

```
<meta name="description" content="Free Web tutorials" />
```

```
<meta name="keywords" content="HTML,CSS,XML,JavaScript" />
```

```
<meta name="author" content="Hege Refsnes" />
```

```
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
```

```
</head>
```

Images

- GIF (Graphic Interchange Format)
 - 8-bit color (256 different colors)
 - JPEG (Joint Photographic Experts Group)
 - 24-bit color (16 million different colors)
 - Both use compression, but **JPEG** compression is **better**
 - Images are inserted into a document with the `` tag with the **src** attribute
 - The **alt** attribute is **required** by XHTML for purposes of:
 1. Non-graphical browsers
 2. Browsers with images turned off
- ``
- The `` tag has 30 different attributes(!), including width and height (in pixels)
 - [Alphabetical Tag List of HTML5](#) (a nice tool to quickly find details of each tag!!)
 - Portable Network Graphics (PNG)
 - Relatively new, big than JPEG files, but not data loss
 - **Should** eventually replace both gif and jpeg

Images (continued): An Example – Image.html

```
<!-- image.html
      An example to illustrate an image    -->
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head> <title> Images </title>
  </head>
  <body>
    <h1> Aidan's Airplanes </h1>
    <h2> The best in used airplanes </h2>
    <h3> "We've got them by the hangarful"
  • <h2> /Special of the month </h2>
  • <p>
    • 1960 Cessna 210 <br />
    • 577 hours since major engine overhaul
    • <br />
    • 1022 hours since prop overhaul
    • <br /><br />
      <img src = "c210new.jpg" alt = "Picture of a Cessna 210"/>
      <br />
      Buy this fine airplane today at a
      remarkably low price <br />
      Call 999-555-1111 today!
    </p>
  </body>
</html>
```


Images (continued)

Aidan's Airplanes

The best in used airplanes

"We've got them by the hangarful"

Special of the month

1960 Cessna 210

577 hours since major engine overhaul

1022 hours since prop overhaul



Buy this fine airplane today at a remarkably low price
Call 999-555-1111 today!

Hypertext Links

- **Hypertext** is the **essence** of the Web!
- A link is specified with the **href** (*hypertext reference*) attribute of **<a>** (the anchor tag)
 - The content of **<a>** is the visual link in the document
 - If the **target** is a whole document (another one), the target need **not** be specified in the target document as being the target
- Note: **Relative addressing** of targets is easier to maintain and more portable than **absolute addressing!**

Hypertext Links: an Example

- <!-- link.html
- An example to illustrate a link -->
- <html xmlns = "<http://www.w3.org/1999/xhtml>">
- <head> <title> Links </title> </head>
- <body>
- <h1> Aidan's Airplanes </h1>
- <h2> The best in used airplanes </h2>
- <h3> "We've got them by the hangarful"
- </h3>
- <h2> Special of the month </h2>
- <p> 1960 Cessna 210

-
Information on the Cessna 210 </p>
</body>
</html>

Hypertext Links (continued)

- If the target is not at the beginning of a doc, it must be marked with a **label**
- **Target labels** can be defined in many different tags with the id attribute, as in

```
<h1 id = "baskets"> Baskets </h1>
```

- The link to an **id** must be preceded by a pound sign (#); if the id is in the same document, this target can simply be

```
<a href = "#baskets"> What about baskets? </a>
```

- If the target is in a different document, the document reference must be included

```
<a href = "myAd.html#baskets"> Baskets </a>
```

- **Style note**: a link should **blend in** with the surrounding text, so *reading without taking the link should not be made less pleasant*

- **Links can be an image** (instead of textual names), for example,

```
<a href = "c210data.html">
```

```
<img src = "smallplane.jpg" alt = "Small picture of an airplane " /> Info on C210  
</a>
```

Lists: 3 types of lists

- **Unordered lists**

- The list is the content of the `` tag

- List elements are the content of the `` tag

```
<h3> Some Common Single-Engine Aircraft </h3>
```

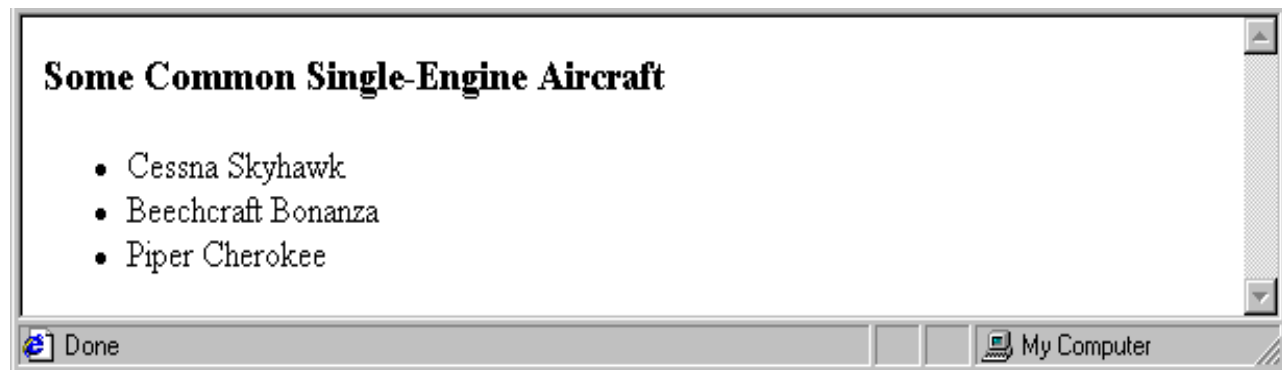
```
<ul>
```

```
  <li> Cessna Skyhawk </li>
```

```
  <li> Beechcraft Bonanza </li>
```

```
  <li> Piper Cherokee </li>
```

```
</ul>
```



- **Ordered lists**

- The list is the content of the `` tag

- Each item in the display is preceded by a sequence value

Lists (continued)

<h3> Cessna 210 Engine Starting Instructions

</h3>

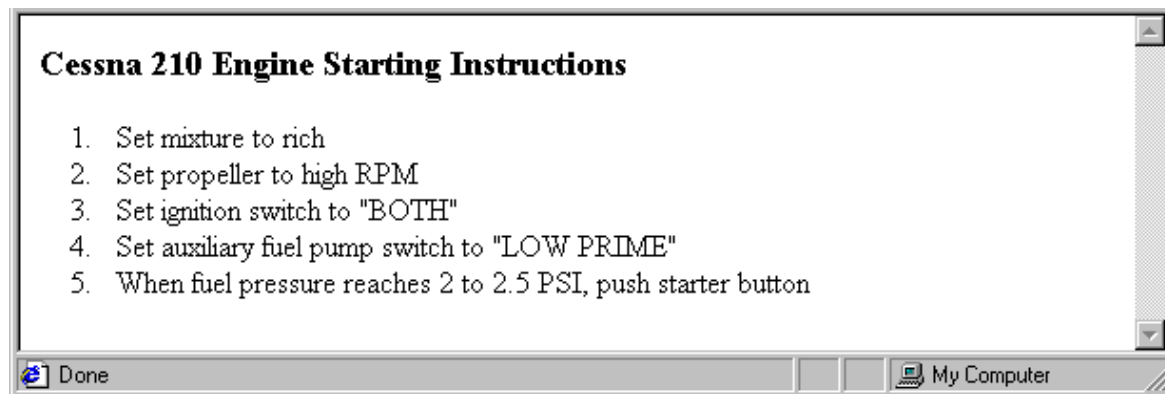
 Set mixture to rich

 Set propeller to high RPM

 Set ignition switch to "BOTH"

 Set auxiliary fuel pump switch to
"LOW PRIME"

 When fuel pressure reaches 2 to 2.5
PSI, push starter button



- *Nested lists*

- Any type of list can be nested inside any type of list
- The nested list must be in a list item

Lists (continued)

- **Definition lists** (for glossaries, etc.)
 - List is the content of the `<dl>` tag
 - Terms being defined are the content of the `<dt>` tag
 - The definitions themselves are the content of the `<dd>` tag

`<h3> Single-Engine Cessna Airplanes </h3>`

`<dl>`

`<dt> 152 </dt>`

`<dd> Two-place trainer </dd>`

`<dt> 172 </dt>`

`<dd> Smaller four-place airplane </dd>`

`<dt> 182 </dt>`

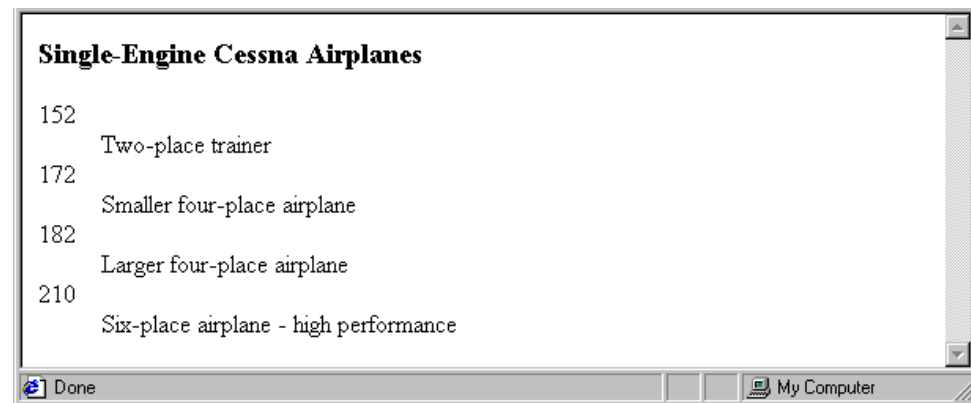
`<dd> Larger four-place airplane </dd>`

`<dt> 210 </dt>`

`<dd> Six-place airplane - high performance`

`</dd>`

`</dl>`



Tables

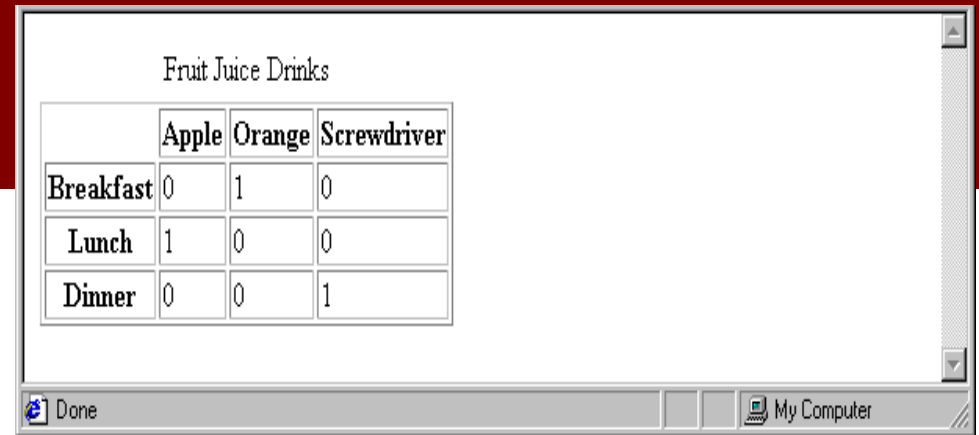
- A table is a matrix of cells, each possibly having content
- The cells can include almost any element
- Some cells have row or column labels and some have data
- A table is specified as the content of a **<table>** tag
- A **border** attribute in the <table> tag specifies a border between the cells
- If border is set to "border", the browser's default width border is used
- The border attr. can be set to a number, which will be the border width
- Without the border attribute, the table will have **no border lines!**
- Tables are given titles with the **<caption>** tag, which must immediately follows <table>

Tables (continued)

- Each row of a table is specified as the content of a `<tr>` tag
- The heading cells are specified as the content of a `<th>` tag
- The data cells are specified as the content of a `<td>` tag

```
<table border = "border">
  <caption> Fruit Juice Drinks </caption>
  <tr>
    <th> </th>
    <th> Apple </th>
    <th> Orange </th>
    <th> Screwdriver </th>
  </tr>
  <tr>
    <th> Breakfast </th>
    <td> 0 </td>
    <td> 1 </td>
    <td> 0 </td>
  </tr>
  <tr>
    <th> Lunch </th>
    <td> 1 </td>
    <td> 0 </td>
    <td> 0 </td>
  </tr>
</table>
```

Tables (continued)



A screenshot of a web browser window displaying a table titled "Fruit Juice Drinks". The table has three columns: "Apple", "Orange", and "Screwdriver". The rows are "Breakfast", "Lunch", and "Dinner". The values in the cells are: Breakfast (Apple: 0, Orange: 1, Screwdriver: 0), Lunch (Apple: 1, Orange: 0, Screwdriver: 0), and Dinner (Apple: 0, Orange: 0, Screwdriver: 1). The browser's status bar at the bottom shows "Done" and "My Computer".

| | Apple | Orange | Screwdriver |
|-----------|-------|--------|-------------|
| Breakfast | 0 | 1 | 0 |
| Lunch | 1 | 0 | 0 |
| Dinner | 0 | 0 | 1 |

- “Supper cells” may take up space of multiple cells
 - – If so, the **colspan** attribute must be set in the `<th>` tag to specify that the label must span some number of columns
- `<tr>`
- `<th colspan = "3"> Fruit Juice Drinks </th>`
- `</tr>`
- `<tr><th> Orange </th>`
 `<th> Apple </th>`
 `<th> Screwdriver </th>`
 `</tr>`



A screenshot of a web browser window displaying a table. The table has a single row with a header cell that spans all three columns, containing the text "Fruit Juice Drinks". Below the header, there are three empty cells. The browser's status bar at the bottom shows "Done" and "My Computer".

| Fruit Juice Drinks | | |
|--------------------|--|--|
| | | |

Tables (continued)

- Similarly the **rowspan** attribute allows covering of multi-rows

```
<table border = "border">
  <tr>
    <td rowspan = "2"> </td>
    <th colspan = "3"> Fruit Juice Drinks
    </th>
  </tr>
  <tr>
    <th> Apple </th>
    <th> Orange </th>
    <th> Screwdriver </th>
  </tr>
  ...
</table>
```

Tables (continued)

- The **align** attribute controls the horizontal placement of cell contents
 - Values are left (**default**), right, and center
 - align is an attribute of <tr>, <th>, and <td> elements
- The **valign** attribute controls the vertical placement
 - Values are top, bottom, and middle or center (**default**)
 - valign is an attribute of <th> and <td> elements

→ The **cellspacing** attribute of <table> is used to specify the distance between cells in a table
- The **cellpadding** attribute of <table> is used to specify the spacing between cell content and the *inner wall* of the cell

Tables (continued)

```
<table cellpadding = "50">
  <tr>
    <td> Colorado is a state of ...
  </td>
    <td> South Dakota is somewhat...
  </td>
  </tr>
</table>
```

| | |
|--|--|
| Colorado is a state of contrasts. The eastern half is a mostly treeless prairie. On the prairie, trees grow only in the Platte and Arkansas river valleys, with a few found along some other small streams. The forested Rocky Mountains rise abruptly from the high plains about midway from east to west and cover most of the western half of the state. There are 54 mountains in Colorado that top 14,000 feet. | South Dakota is somewhat similar to Colorado in that it is a mostly treeless prairie in the east, but has a range of forested mountains in the west. But in South Dakota, the mountains, named the Black Hills, lie only in the far western part of the state and rise to only a little over 7500 feet. However, they are still the highest mountains east of the Rockies in the U.S. The famous Mount Rushmore is nestled in the middle of the Black Hills. |
|--|--|

Table (continued): New Features in HTML5

- *Table Sections*

- Header, body, and footer, which are the elements of table: `thead`, `tbody`, and `tfoot`
- If a document has *multiple* `tbody` elements, they are separated by thicker horizontal lines

- *Uses of Tables*

- In the past, tables were used to align elements in rows and columns – for general (page) layout
- That use of tables is now frowned upon!
- Nowadays the pros tend to use Cascading Style Sheets to place elements in rows and columns – as general layout
- Use tables only when the information is naturally tabular!

Forms

- A form is the *usual* way for server to receive data from browser
- HTML has tags to create a collection of objects that accomplish this information gathering
 - The objects are called *widgets* (e.g., radio buttons and checkboxes)
- When the **Submit** button of a form is clicked, the form's values are sent to the server
- All of the widgets are defined in the content of a **<form>** tag
 - The only **required** attribute of **<form>** is **action**, which specifies the URL of the application that is to be called when the *Submit* button is clicked
- E.g.: **action = "<http://www.cs.ucp.edu/survey.php>"**
 - » If the form has no action, the value of action is an empty string

Forms (continued)

- The **method** attribute of `<form>` specifies the method of transferring the form data to the server, such as **get** and **post**
- The **enctype** attribute
 - As a quick example, the following form-data will be sent unencoded:
`<form action="form_action.asp" method="get" enctype="text/plain">`
First name: `<input type="text" name="fname" />
`
Last name: `<input type="text" name="lname" />
`
`<input type="submit" value="Submit" />`
`</form>`
 - The **enctype** attribute specifies how form-data should be **encoded** before sending it to the server.
 - By **default**, `enctype="application/x-www-form-urlencoded"`. This means that all characters are encoded before they are sent (spaces are converted to "+" symbols, and special characters converted to **ASCII Hexadecimal values**).

Forms (continued)

Widgets

– **Many** are created with the `<input>` tag

» The `type` attribute of `<input>` specifies the kind of widget being created

1. Text

- Creates a horizontal **text box** for text input
- Default size is 20; it can be changed with the **size** attribute
- If more characters entered than will fit, the box is **scrolled** (shifted) left
- If you don't allow the user to type more characters than will fit, set **maxlength**, which causes excess input to be **ignored**

```
<input type = "text" name = "Phone" size = "12" >
```

» ***Password*** – just like text except asterisks are displayed, rather than the input characters

Forms (continued)

2. **Checkboxes** - to collect multiple choice input

- » Every checkbox requires a `value` attribute, which is the widget's value in the form data when the checkbox is 'checked'
 - A unchecked checkbox contributes no value to the form data
- » By default, no checkbox is initially 'checked'
- » To initialize a checkbox to 'checked', the **checked** attribute must be set to `"checked"`

[**example** on next slide]

Forms (continued)

Grocery Checklist

```
<form action = ">  
  <p>  
    <input type = "checkbox"  name ="groceries"  
          value = "milk"  checked = "checked">  
      Milk  
    <input type = "checkbox"  name ="groceries"  
          value = "bread">  
      Bread  
    <input type = "checkbox"  name = "groceries"  
          value= "eggs">  
      Eggs  
  </p>  
</form>
```

Grocery Checklist

☒ Milk ☐ Bread ☐ Eggs

Forms (continued)

3. *Radio Buttons* - collections of checkboxes in which **only one** button can be 'checked' at a time
 - Every button in a radio button **group** MUST have the **same name**
- **Example** on next slide

Forms (continued)

3. Radio Buttons (continued)

- If no button in a group is 'pressed', the browser **often** 'presses' the first one

Age Category

```
<form action = ">
```

```
<p>
```

```
<input type = "radio"  name = "age"  
  value = "under20" checked = "checked"> 0-19
```

```
<input type = "radio"  name = "age"  
  value = "20-35"> 20-35
```

```
<input type = "radio"  name = "age"  
  value = "36-50"> 36-50
```

```
<input type = "radio"  name = "age"  
  value = "over50"> Over 50
```

```
</p>
```

```
</form>
```



Age Category

☒ 0-19 ☐ 20-35 ☐ 36-50 ☐ Over 50

Forms (continued)

4. Menus - created with `<select>` tag

- There are two kinds of menus, those that behave like *checkboxes* and those that behave like *radio buttons* (the **default**)
 - Checkboxes-like menus must include the **multiple** attribute set to "multiple"
- The **name** attribute of `<select>` is required
- The **size** attribute of `<select>` can be included to specify the number of menu items to be displayed (the default is 1)
 - If `size > 1` or `multiple` is specified, the menu is displayed as a “**pop-up**” menu (otherwise “**drop-down**”)

Forms (continued)

4. Menus (continued)

- Each **item** of a menu is specified with an `<option>` tag, whose pure text content is the value of the item
- An `<option>` tag can include the `selected` attribute, which when assigned "selected" specifies that the item is **pre-selected**

Grocery Menu - milk, bread, eggs, cheese

```
<form action = "">
  <p>
    With size = 1 (the default)
    <select name = "groceries">
      <option> milk </option>
      <option> bread </option>
      <option> eggs </option>
      <option> cheese </option>
    </select>
  </p>
</form>
```

Forms (continued)

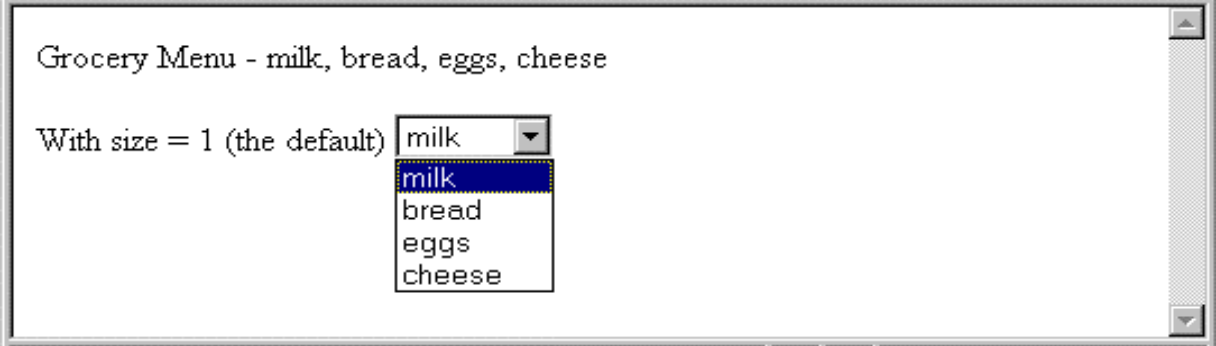
- *Widgets (continued)*



Grocery Menu - milk, bread, eggs, cheese

With size = 1 (the default) milk


- After clicking the menu:



Grocery Menu - milk, bread, eggs, cheese

With size = 1 (the default) milk
milk
bread
eggs
cheese

- After changing `size` to 2:



Grocery Menu - milk, bread, eggs, cheese

With size = 2 (specified) milk
bread

Forms (continued)

5. Text areas - created with `<textarea>`

- Usually for bigger chunk of text, using `rows` and `cols` attributes to specify the size of the text area
- **Default text** can be included as the content of `<textarea>`
- **Scrolling** is implicit if the area is overfilled

Please provide your employment aspirations

```
<form action = "">
```

```
<p>
```

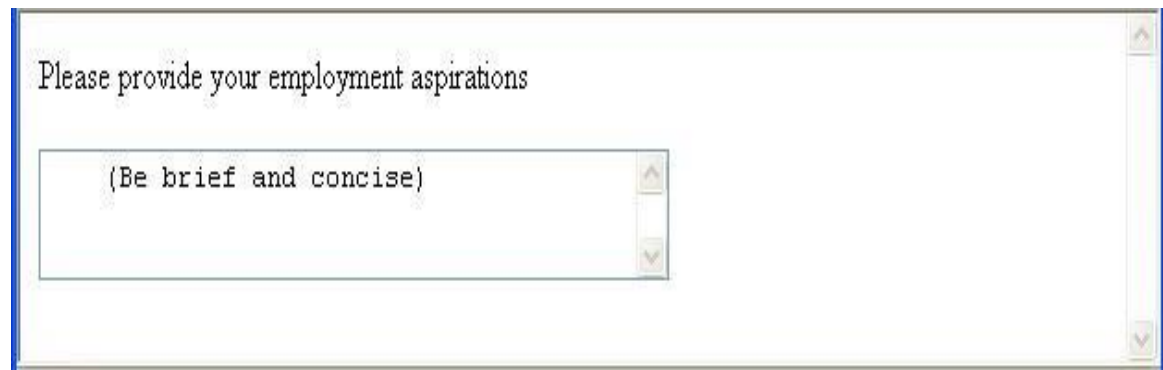
```
<textarea name = "aspirations"  rows = "3"  
          cols = "40">
```

```
(Be brief and concise)
```

```
</textarea>
```

```
</p>
```

```
</form>
```



Forms (continued)

6. Reset and Submit buttons

- Both are created with `<input>`

```
<input type = "reset" value = "Reset Form">
```

```
<input type = "submit" value = "Submit Form">
```

Submit has two actions:

1. Encodes the data of the form
 2. Requests that the server to execute the server-resident program specified as the value of the `action` attribute of `<form>`
- A Submit button is required in every form

Next we will introduce a few new tags added by HTML5

HTML5 ADDED TAGS

HTML5 (continued)

- The `audio` Element

- Prior to HTML5, a plug-in was required to play sound while a document was being displayed
- Audio encoding algorithms are called *audio codecs* – e.g., MP3, Vorbis
- Coded audio data is stored in containers—e.g., Ogg, MP3, and Wav (file name extension indicates the container, not the audio code)
- Vorbis code is stored in Ogg containers
- MP3 code is stored in MP3 containers
- Wav code is stored in Wav containers

HTML5 (continued)

- The `audio` Element (continued)

- General syntax:

`<audio attributes>`

`<source src = "filename1" >`

`...`

`<source src = "filenamen" >`

`Your browser does not support the audio element`

`</audio>`

- Browser chooses the first audio file it can play and skips the content; if none, it displays the content
- Different browsers have different audio capabilities
- The *controls* attribute, which is set to "`controls`", creates a start/stop button, a clock, a progress slider, total time of the file, and a volume slider

HTML5 (continued)

- The `video` Element

- Prior to HTML5, there was no standard way to play video clips while a document was being displayed
- Video **codecs** are stored in containers
- Video codecs:
 - H.264 (MPEG-4 AVC) – can be stored in an MPEG-4 container
 - Theora – can be stored in any container
 - VP8—can be stored in any container

HTML5 (continued)

- Different browsers support different video codecs
- The `width` and `height` attributes set the screen size
- The `autoplay` attribute, set to "`autoplay`", specifies that the video should play as soon as it is ready
- The `preload` attribute, set to "`preload`", specifies that the video should be loaded as soon as the document is loaded
- The `controls` attribute, set to "`controls`", is like that of the audio element

HTML5 (continued)

- Organizational Elements

- Header Elements

- hgroup – a **container** for header information (within <body> elem)

```
<hgroup>
```

```
  <header>
```

```
    <h1> The Podunk Press </h1>
```

```
    <h2> "All the news we can fit" </h2>
```

```
  </header>
```

```
  -- table of contents --
```

```
</hgroup>
```


HTML5 (continued)

- **Footer Elements**

- **footer** – a container for footer information (within <body> elem)

```
<footer>
```

```
&copy; The Podunk Press, 2012
```

```
<br />
```

```
Editor in Chief: Squeak Martin
```

```
</footer>
```

- The **section** Element – a container for sections
 - The **article** Element – a container for self-contained part of a document (from another source)
 - The **aside** Element – a container for tangential info
 - The **nav** Element – navigation sections (list of links)

Example of <nav>

```
<nav>  
  <a href="/html/">HTML</a> |  
  <a href="/css/">CSS</a> |  
  <a href="/js/">JavaScript</a> |  
  <a href="/jquery/">jQuery</a>  
</nav>
```

HTML5 (continued)

- The `time` Element

- For putting a time stamp on a document
- Two parts, text and machine-readable (`datetime`)
- The machine-readable part: given by the `datetime` attribute (optional) –
 - Date part: 4-digit year, a dash, 2-digit month, a dash, 2-digit day of the month ("2012-08-29")
 - Time (optional) format: T09:00
- The text part: given as the content of `<time>`

```
<time datetime = "2012-08-29T09:00">
  August 8, 2012 9:00 am
</time>
```
- The two parts need not specify the same date
- Deficiencies:
 1. Dates prior to the Christian era are not possible
 2. No approximations

Syntactic Differences: XHTML vs. HTML

- **Case sensitivity** (see notes page)
- **Closing tags**
- **Quoted attribute values**
- **Explicit attribute values**
- **id and name attributes**
- **Element nesting**