WEB TECHNOLOGIES
A COMPUTER SCIENCE PERSPECTIVE

JEFFREY C. JACKSON

Chapter 2
Markup Languages:
XHTML 1.0

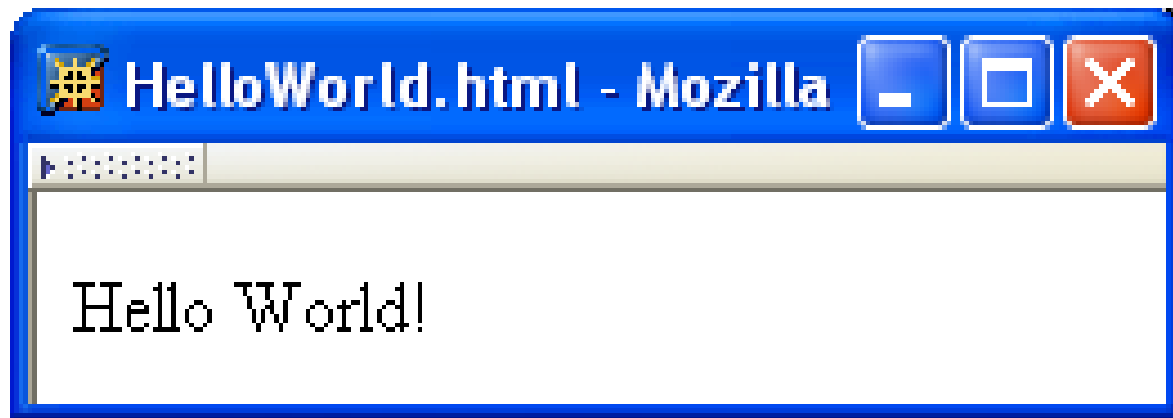# HTML "Hello World!"

Document
Type
Declaration

```
<!DOCTYPE html
        PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>
      HelloWorld.html
    </title>
  </head>
  <body>
    <p>
      Hello World!
    </p>
  </body>
</html>
```
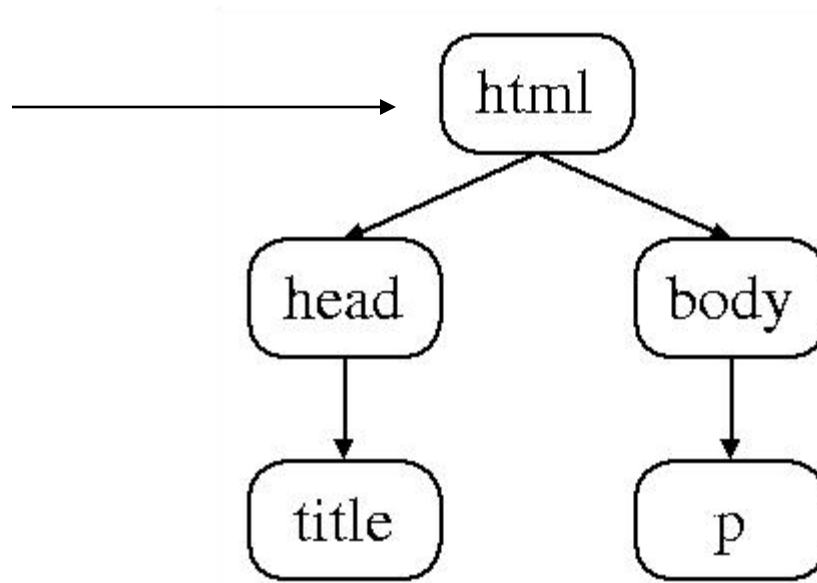
Document
Instance

HTML "Hello World"

HTML Tags and Elements

- Any string of the form < … > is a *tag*
- All tags in document instance of Hello World are either end tags (begin with </) or start tags (all others)
  - Tags are an example of markup, that is, text treated specially by the browser
  - Non-markup text is called character data and is normally displayed by the browser
- String at beginning of start/end tag is an element name
- Everything from start tag to matching end tag, including tags, is an element
  - Content of element excludes its start and end tags

HTML Element Tree

Root
Element

HTML Root Element

- Document type declaration specifies name of root element:
  `<!DOCTYPE html`

- Root of HTML document must be `html`

- XHTML 1.0 (standard we will follow) requires that this element contain `xmlns` attribute specification (name/value pair)

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

HTML head and body Elements

- The body element contains information displayed in the browser client area
- The head element contains information used for other purposes by the browser:
  - title (shown in title bar of browser window)
  - scripts (client-side programs)
 - style (display) information
  - etc.

HTML History

- 1990: HTML invented by Tim Berners-Lee
- 1993: Mosaic browser adds support for images, sound, video to HTML
- 1994-~1997: "Browser wars" between Netscape and Microsoft, HTML defined operationally by browser support
- ~1997-present: Increasingly, World-Wide Web Consortium (W3C) recommendations define HTML

HTML Versions

- HTML 4.01 (Dec 1999) syntax defined using Standard Generalized Markup Language (SGML)

- XHTML 1.0 (Jan 2000) syntax defined using Extensible Markup Language (XML)

- Primary differences:
  - HTML allows some tag omissions (e.g., end tags)
  - XHTML element and attribute names are lower case (HTML names are case-insensitive)
 - XHTML requires that attribute values be quoted

SGML and XML
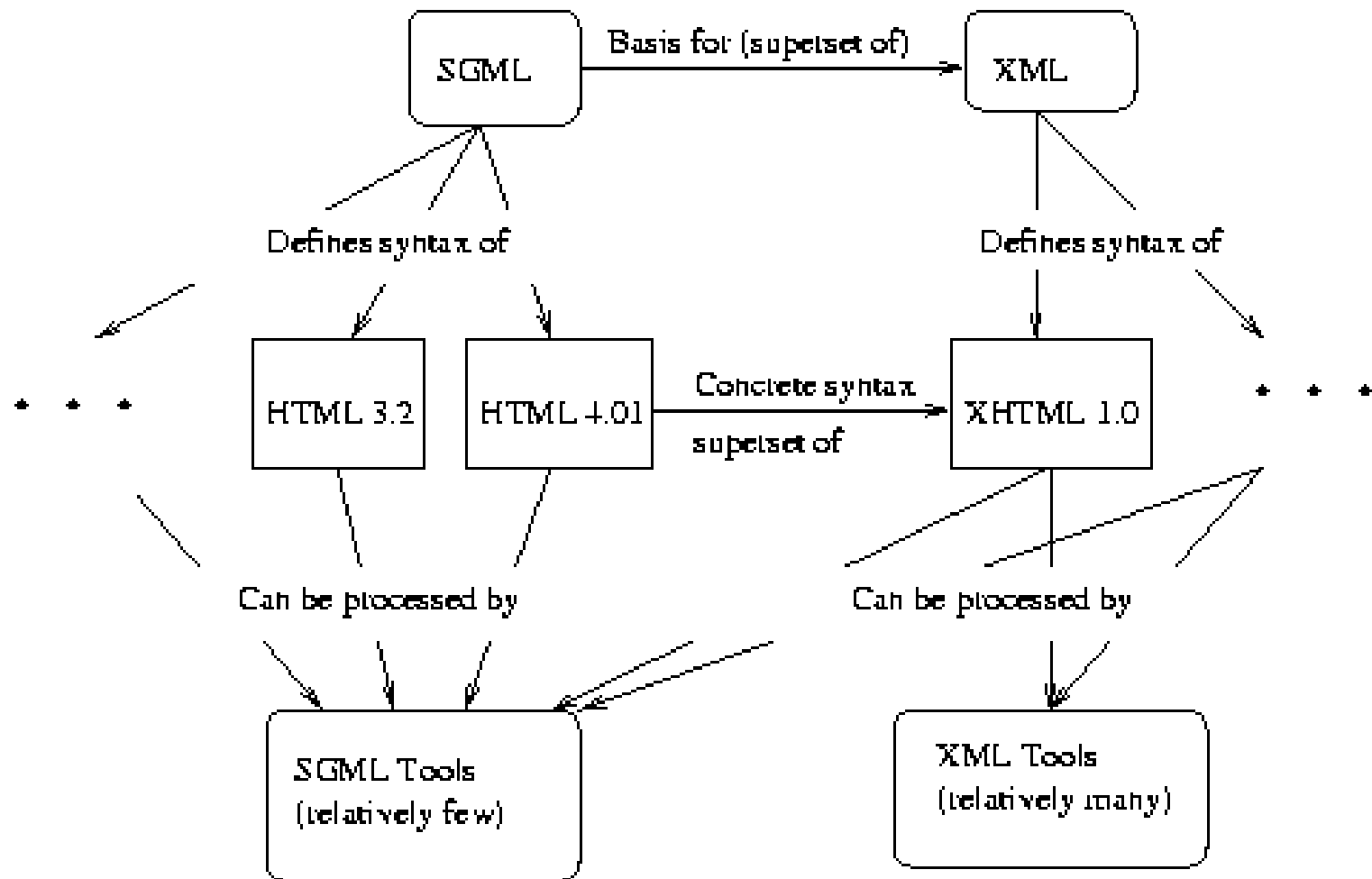
HTML "Flavors"

- For HTML 4.01 and XHTML 1.0, the document type declaration can be used to select one of three "flavors":
  - Strict: W3C ideal
  - Transitional: Includes deprecated elements and attributes (W3C recommends use of *style sheets* instead)
  - Frameset: Supports frames (subwindows within the client area)

HTML Frameset
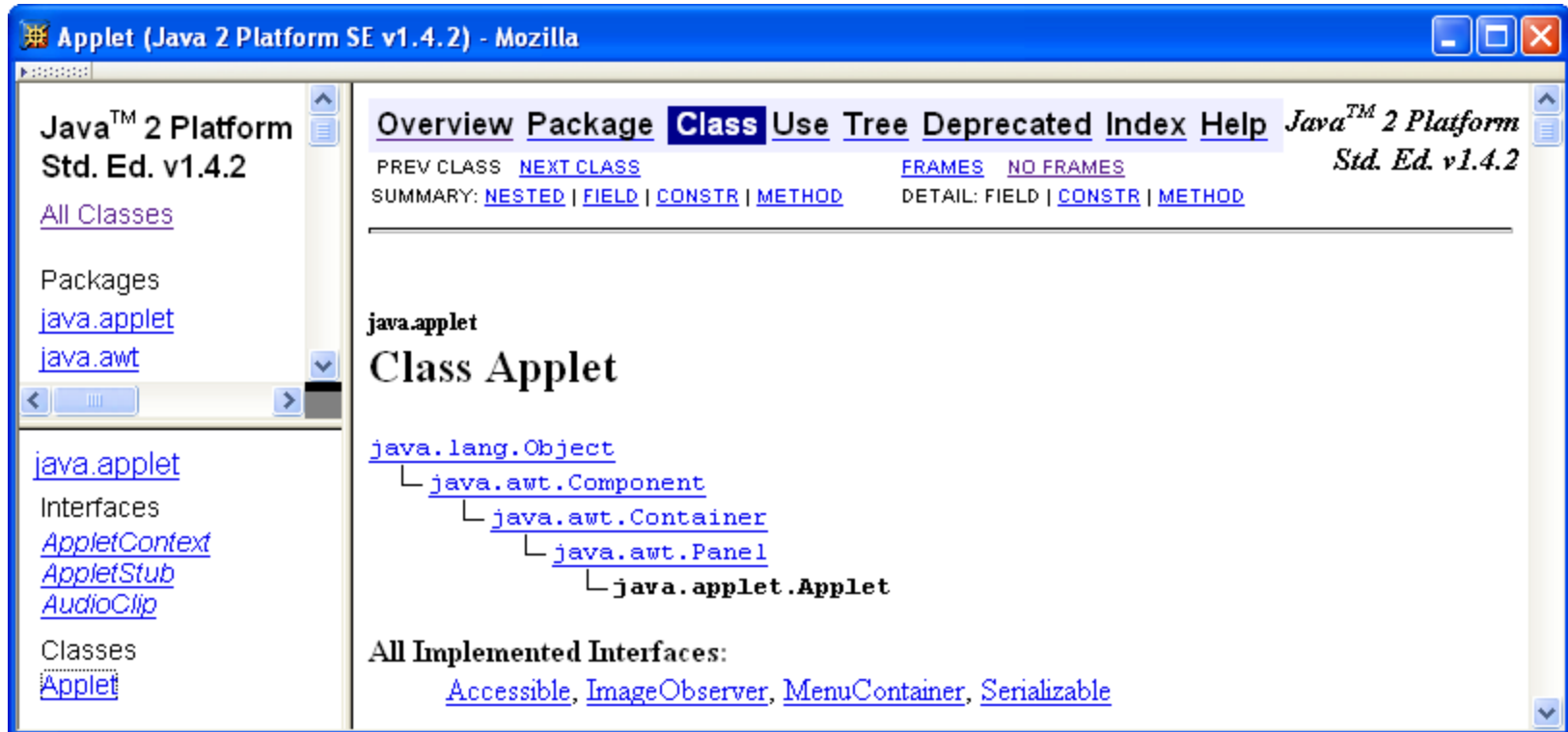
Screen shots are reproduced by permission of Sun Microsystems Inc. All rights reserved.

HTML Document Type Declarations

- XHTML 1.0 Strict:
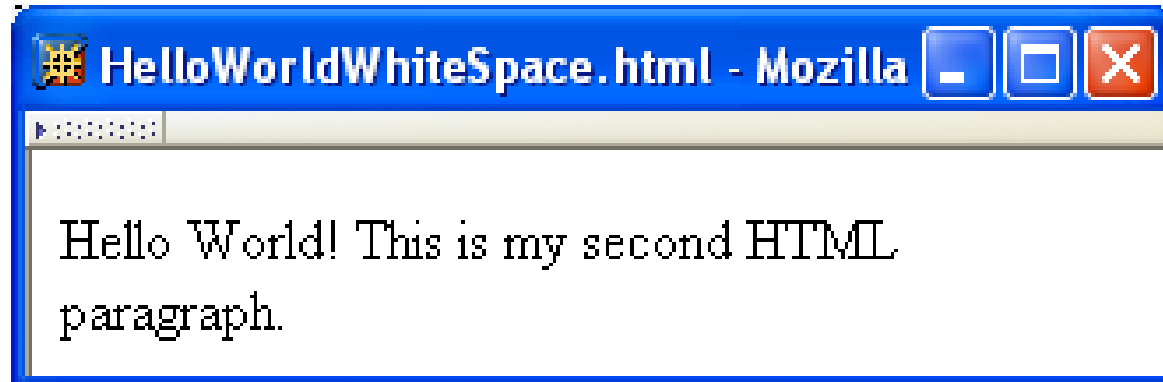 <!DOCTYPE html
  PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

- XHTML 1.0 Frameset:
  <!DOCTYPE html
  PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

- HTML 4.01 Transitional:
  <!DOCTYPE HTML
  PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">

XHTML White Space

- Four white space characters: carriage return, line feed, space, horizontal tab
- Normally, character data is normalized:
  - All white space is converted to space characters
  - Leading and trailing spaces are trimmed
  - Multiple consecutive space characters are replaced by a single space character

XHTML White Space

```
<body>
  <p>
    Hello World!

    This is my second HTML paragraph.
  </p>
</body>
```
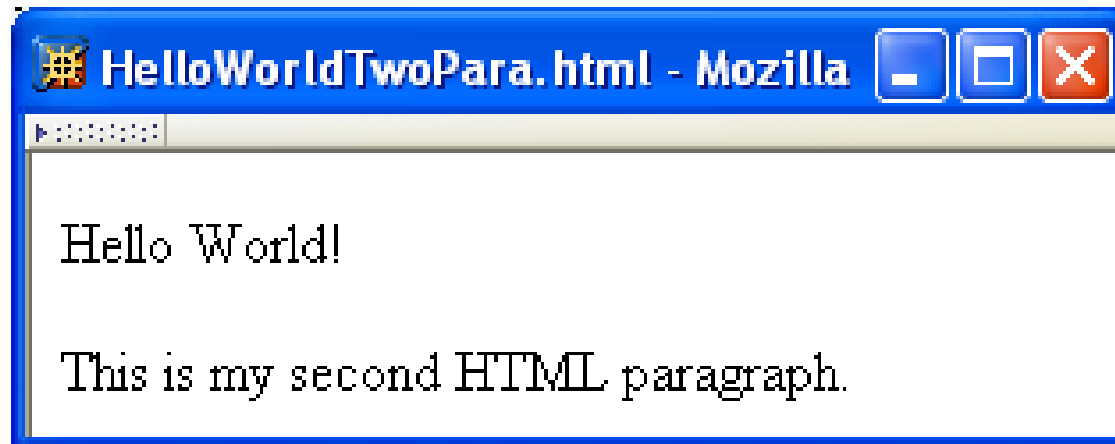

HelloWorldWhiteSpace.html - Mozilla

Hello World! This is my second HTML paragraph.

XHTML White Space

```
<p>
   Hello World!
</p>
<p>
   This is my second HTML paragraph.
</p>
```


Hello World!

This is my second HTML paragraph.

Unrecognized HTML Elements

```
<!DOCTYPE html
        PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <titl>
       HelloWorldBadElt.html
    </title>
  </head>
  <body>
    <p>
       Hello World!
    </p>
  </body>
</html>
```

Misspelled element name →

Unrecognized HTML Elements

Belongs
here

`title` character
data

HelloWorldBadElt.html

Hello World!

Unrecognized HTML Elements

`title` character data

Displayed here

Unrecognized HTML Elements

- Browsers ignore tags with unrecognized element names, attribute specifications with unrecognized attribute names
  - Allows evolution of HTML while older browsers are still in use
- Implication: an HTML document may have errors even if it displays properly
- Should use an <u>HTML validator</u> to check syntax

HTML References

- Since < marks the beginning of a tag, how do you include a < in an HTML document?

- Use markup known as a reference

- Two types:

  - Character reference specifies a character by its Unicode code point

    - For <, use `<` or `<` or `<`

  - Entity reference specifies a character by an HTML-defined name

    - For <, use `<`

HTML References

TABLE 2.2: Example entity and character references.

| Character	Entity Reference	Character Reference (decimal)
<	<	<
>	>	>
&	&	&
"	"	"
'	'	'
©	©	©
ñ	ñ	ñ
α	α	α
$\forall$	∀	∀

HTML References

- Since `<` and `&` begin markup, within character data or attribute values these characters must ***always*** be represented by references (normally `<` and `&`)
- Good idea to represent `>` using reference (normally `>`)
  - Provides consistency with treatment of <
  - Avoids accidental use of the reserved string]]>

HTML References

- Non-breaking space (` `) produces space but counts as part of a word
  - Ex: keep together keep together …
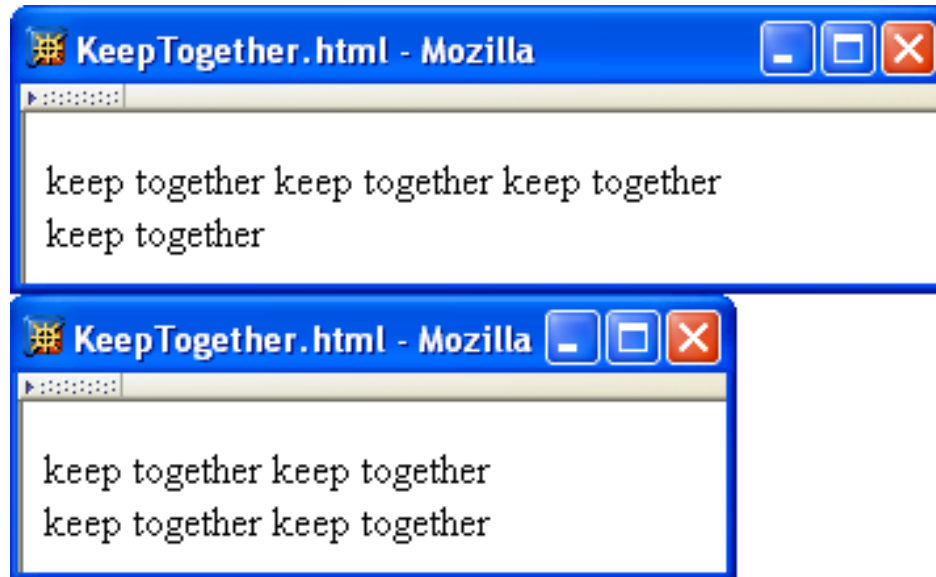
HTML References

- Non-breaking space often used to create multiple spaces (not removed by normalization)

```
<p>
  Hey, you.  Yes.  I am talking to you.
</p>
<p>
  Hey, you.  Yes.  I am talking to you.
</p>
```

  + space displays as two spaces

Sentences.html - Mozilla

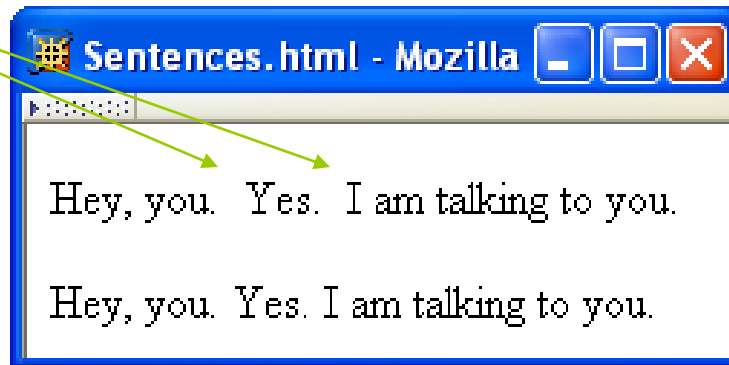Hey, you. Yes. I am talking to you.

Hey, you. Yes. I am talking to you.
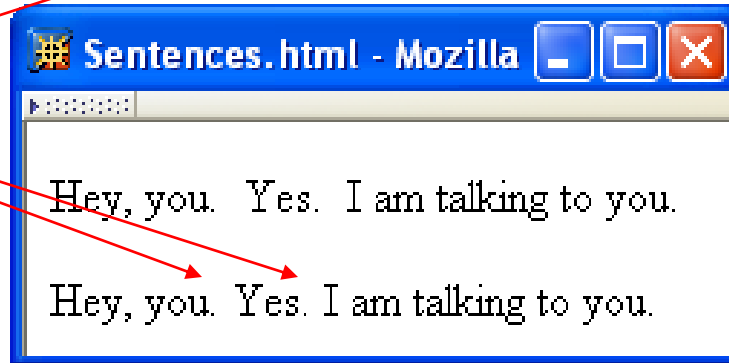
HTML References

- Non-breaking space often used to create multiple spaces (not removed by normalization)

```
<p>
  Hey, you.  Yes.  I am talking to you.
</p>
<p>
  Hey, you.  Yes.  I am talking to you.
</p>
```

two spaces
display as one

Sentences.html - Mozilla

Hey, you. Yes. I am talking to you.

Hey, you. Yes. I am talking to you.

XHTML Attribute Specifications

- Example:

```
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
```
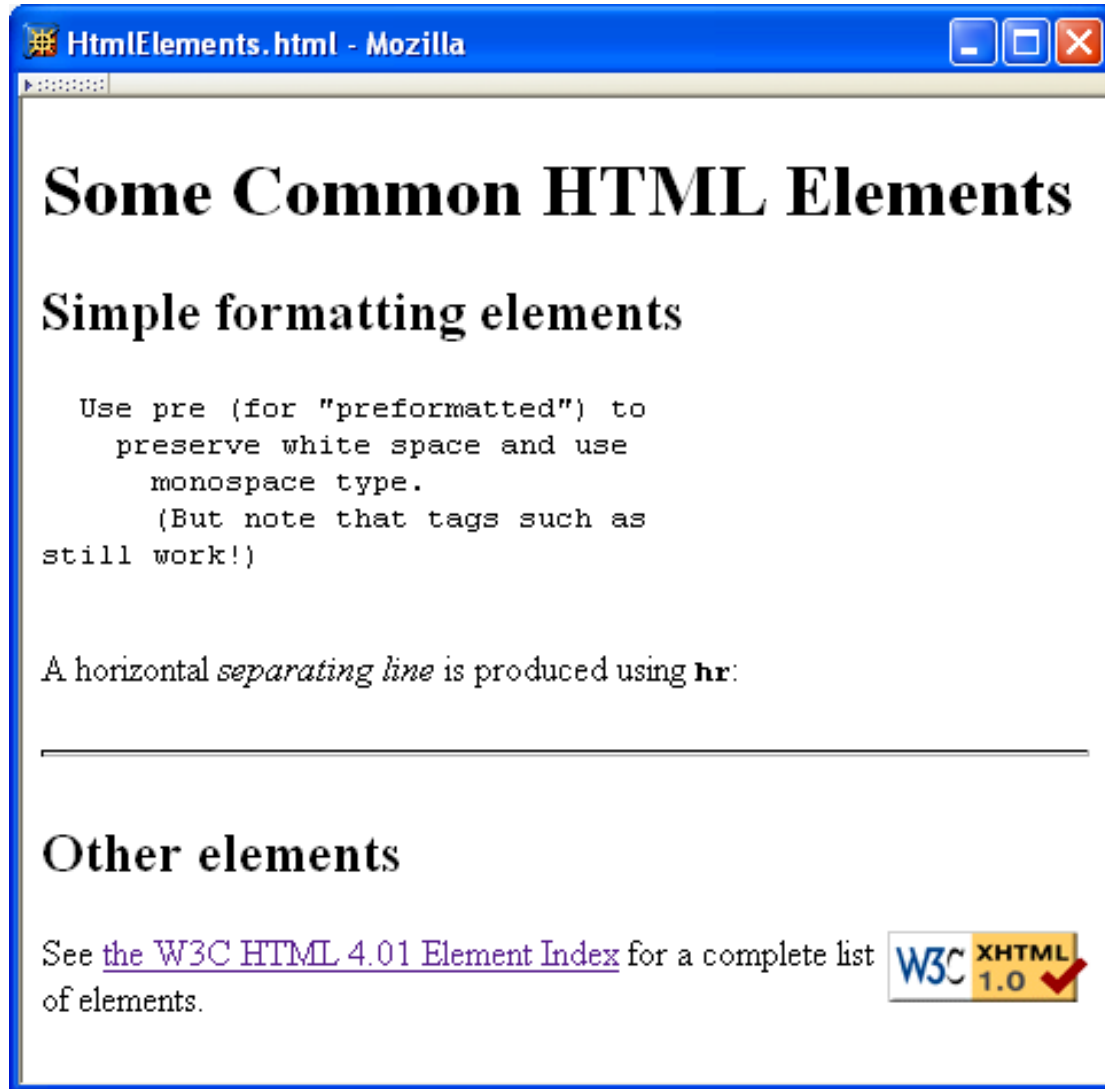
- Syntax:
  - Valid attribute names specified by HTML recommendation (or XML, as in xml:lang)
  - Attribute values must be quoted (matching single or double quotes)
  - Multiple attribute specifications are space-separated, order-independent

XHTML Attribute Values

- Can contain embedded quotes or references to quotes

  ✓ `value = "Ain't this grand!"`
  ✓ `value = "He said, "She said", then sighed."`
  ✗ `value = "He said, "She said", then sighed."`

- May be normalized by browser

  – Best to normalize attribute values yourself for optimal browser compatibility
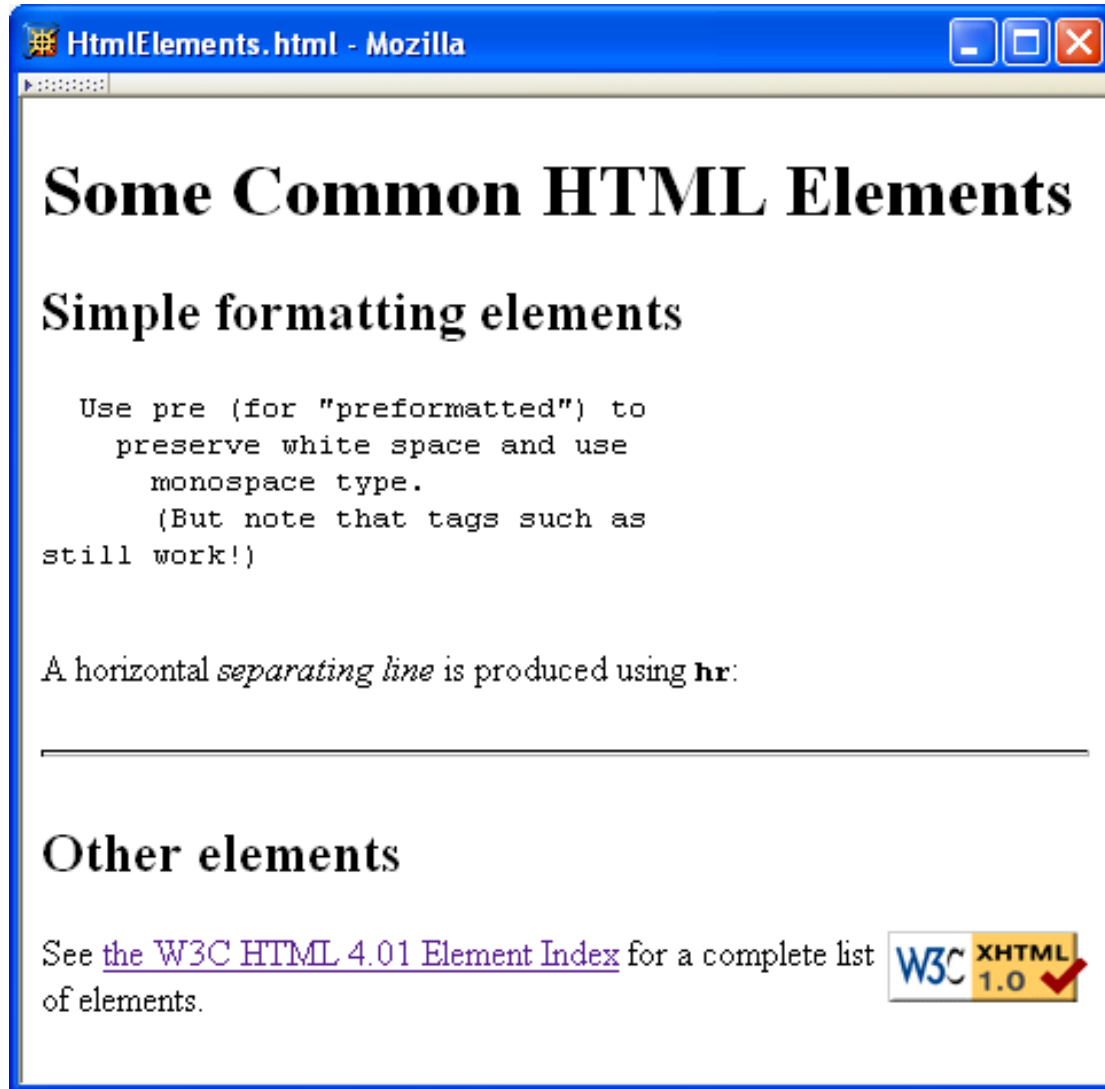
Common HTML Elements

HtmlElements.html - Mozilla

Some Common HTML Elements

Simple formatting elements

```
   Use pre (for "preformatted") to
     preserve white space and use
       monospace type.
       (But note that tags such as
still work!)
```

A horizontal *separating line* is produced using **hr**:

Other elements

See the W3C HTML 4.01 Element Index for a complete list of elements.

Common HTML Elements

- Headings are produced using h1, h2, …, h6 elements:

```
<h1>
   Some Common HTML Elements
</h1>
<h2>
   Simple formatting elements
</h2>
```

- Should use h1 for highest level, h2 for next highest, etc.
  - Change style (next chapter) if you don't like the "look" of a heading

Common HTML Elements

HtmlElements.html - Mozilla

Some Common HTML Elements

Simple formatting elements

```
   Use pre (for "preformatted") to
     preserve white space and use
       monospace type.
       (But note that tags such as
still work!)
```

A horizontal *separating line* is produced using **hr**:

Other elements

See the W3C HTML 4.01 Element Index for a complete list of elements.

Common HTML Elements

- Use `pre` to retain format of text and display using monospace font:

```
<pre>
Use pre (for "preformatted") to
  preserve white space and use
    monospace type.
    (But note that tags such as<br />still work!)
</pre>
```
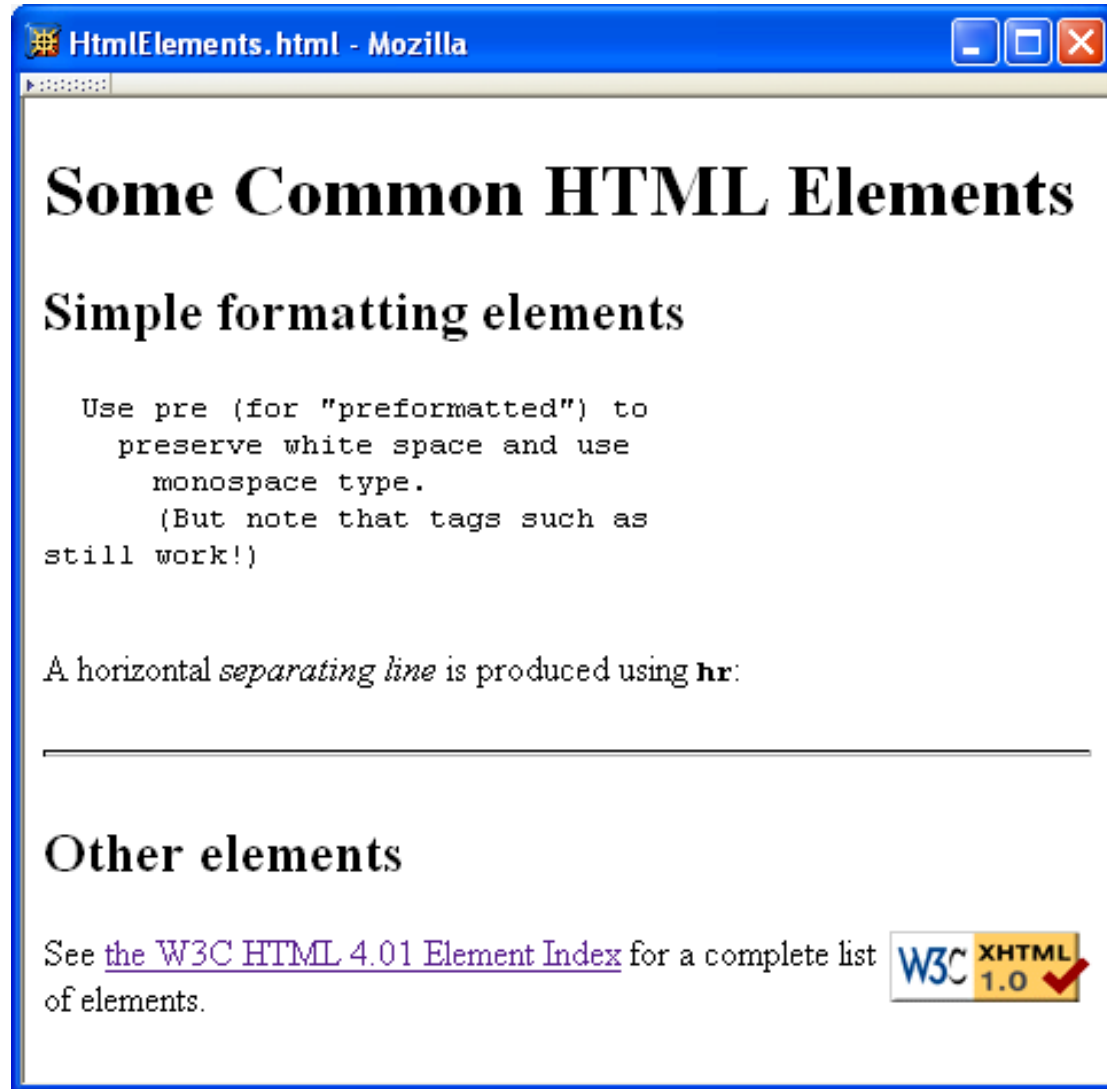
- Note that any embedded markup (such as `<br />`) is still treated as markup!

Common HTML Elements

- `br` element represents line break
- `br` is example of an empty element, i.e., element that is not allowed to have content
- XML allows two syntactic representations of empty elements
  - Empty tag syntax <br /> is recommended for browser compatibility
  - XML parsers also recognize syntax
</br> (start tag followed immediately by end tag), but many browsers do not understand this for empty elements

Common HTML Elements

HtmlElements.html - Mozilla

Some Common HTML Elements

Simple formatting elements

```
   Use pre (for "preformatted") to
     preserve white space and use
       monospace type.
       (But note that tags such as
still work!)
```

A horizontal *separating line* is produced using **hr**:

Other elements

See the W3C HTML 4.01 Element Index for a complete list of elements.

Common HTML Elements

- Text can be formatted in various ways:
  - Apply style sheet technology (next chapter) to a span element (a styleless wrapper):

    ```
    <span style="font-style:italic">separating line</span>
    ```

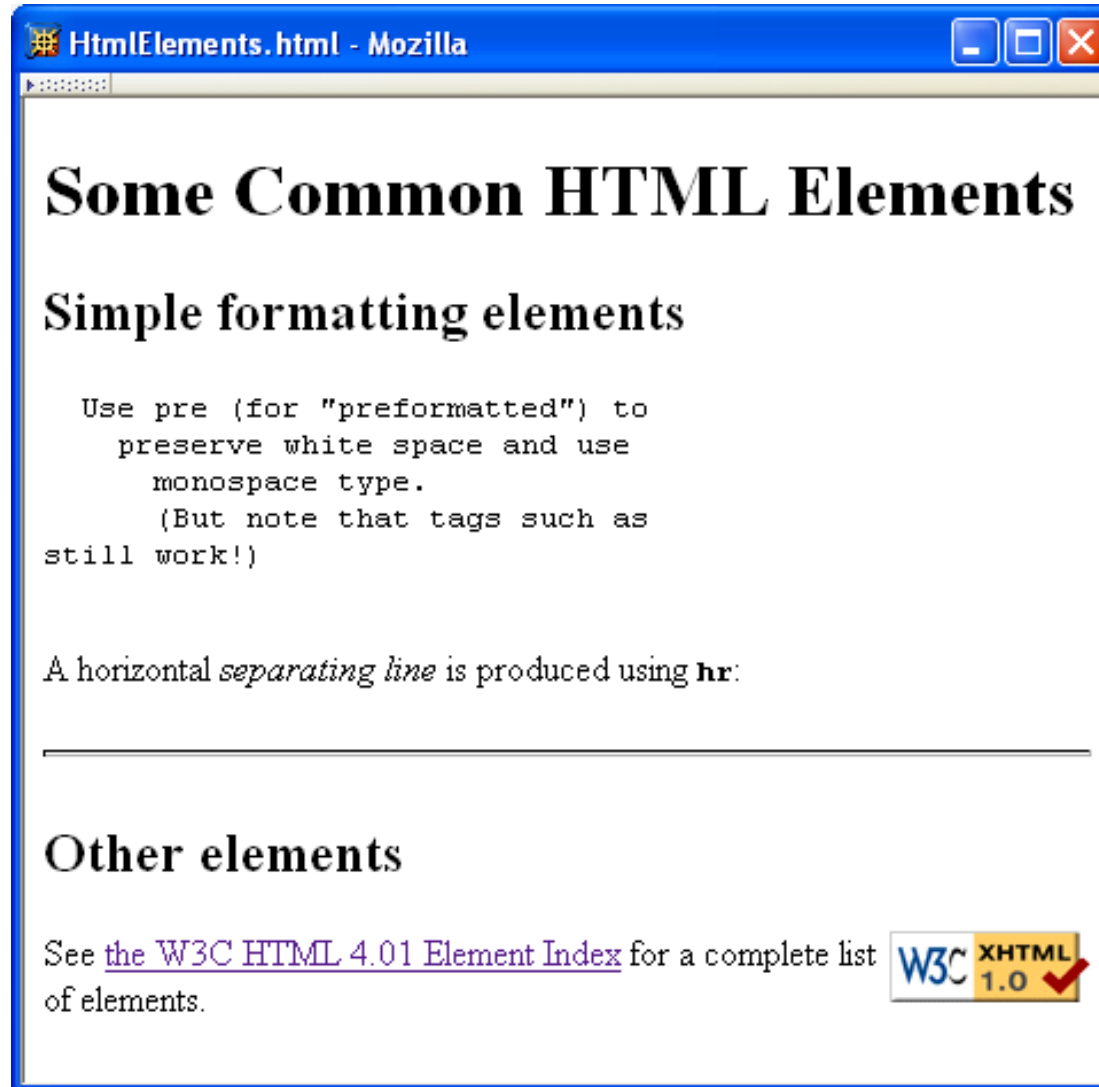 - Use a phrase element that specifies semantics of text (not style directly):

    ```
    <strong>hr</strong>
    ```

  - Use a font style element
    - Not recommended, but frequently used

Common HTML Elements

TABLE 2.3: HTML font style elements.

Element	Font used by content
b	Bold-face
i	Italic
tt	"Teletype" (fixed-width font)
big	Increased font size
small	Decreased font size

Common HTML Elements

HtmlElements.html - Mozilla

Some Common HTML Elements

Simple formatting elements

```
   Use pre (for "preformatted") to
     preserve white space and use
        monospace type.
        (But note that tags such as
still work!)
```

A horizontal *separating line* is produced using **hr**:

Other elements

See the W3C HTML 4.01 Element Index for a complete list of elements.

W3C XHTML 1.0

Common HTML Elements

- Horizontal rule is produced using hr
- Also an empty element
- Style can be modified using style sheet technology

Common HTML Elements

HtmlElements.html - Mozilla

Some Common HTML Elements

Simple formatting elements

```
   Use pre (for "preformatted") to
     preserve white space and use
        monospace type.
        (But note that tags such as
still work!)
```

A horizontal *separating line* is produced using **hr**:

Other elements

See the W3C HTML 4.01 Element Index for a complete list of elements.

W3C XHTML 1.0
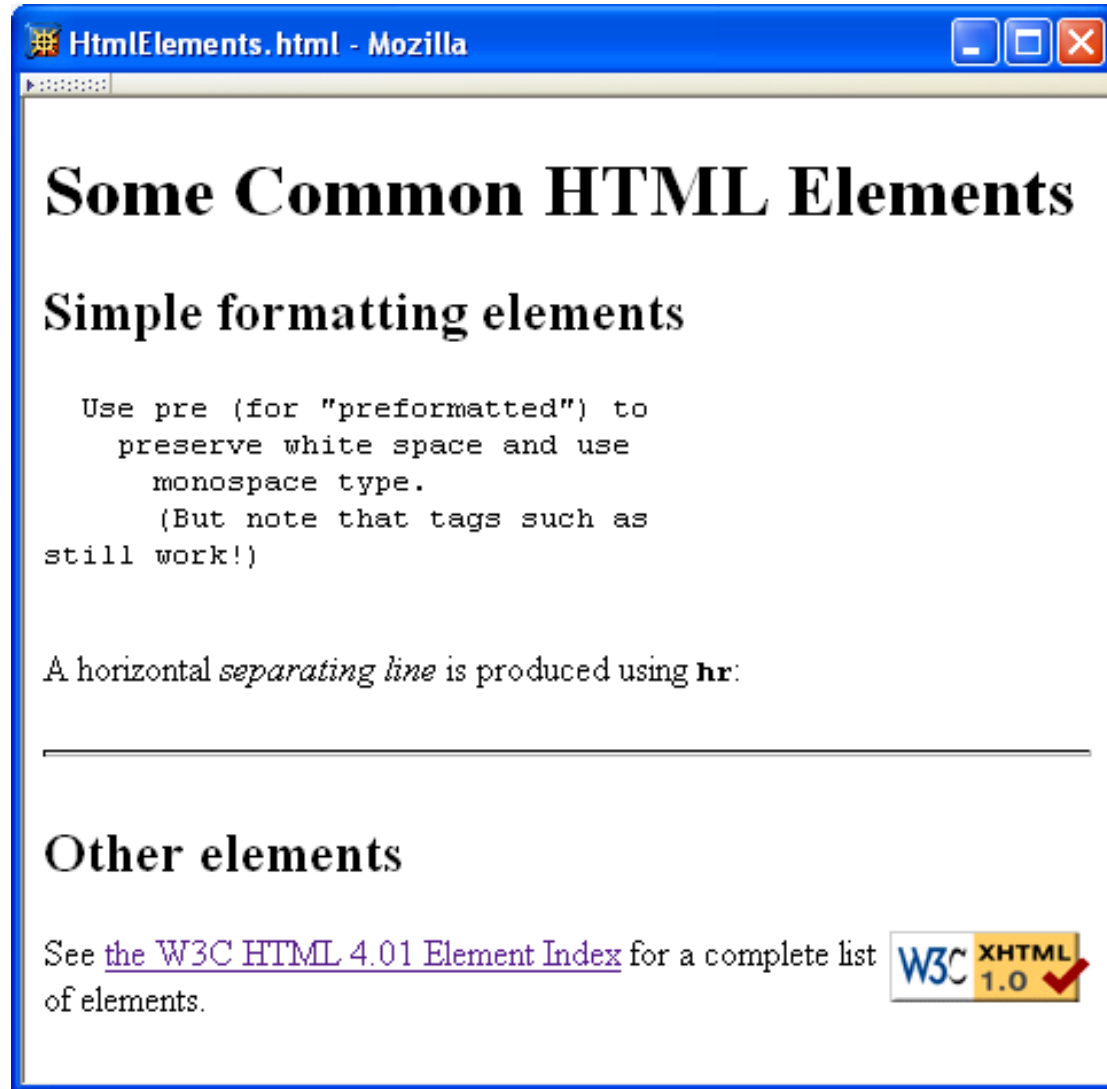
Common HTML Elements

- Images can be embedded using `img` element

```
<img
    src="http://www.w3.org/Icons/valid-xhtml10"
    alt="Valid XHTML 1.0!" height="31" width="88"
    style="float:right" />
```

- Attributes:
  - `src`: URL of image file (required). Browser generates a GET request to this URL.
  - `alt`: text description of image (required)
  - `height`/`width`: dimensions of area that image will occupy (recommended)

Common HTML Elements

- If height and width not specified for image, then browser may need to rearrange the client area after downloading the image (poor user interface for Web page)

- If height and width specified are not the same as the original dimensions of image, browser will resize the image

- Default units for height and width are "picture elements" (pixels)
  - Can specify percentage of client area using string such as "50%"

Common HTML Elements
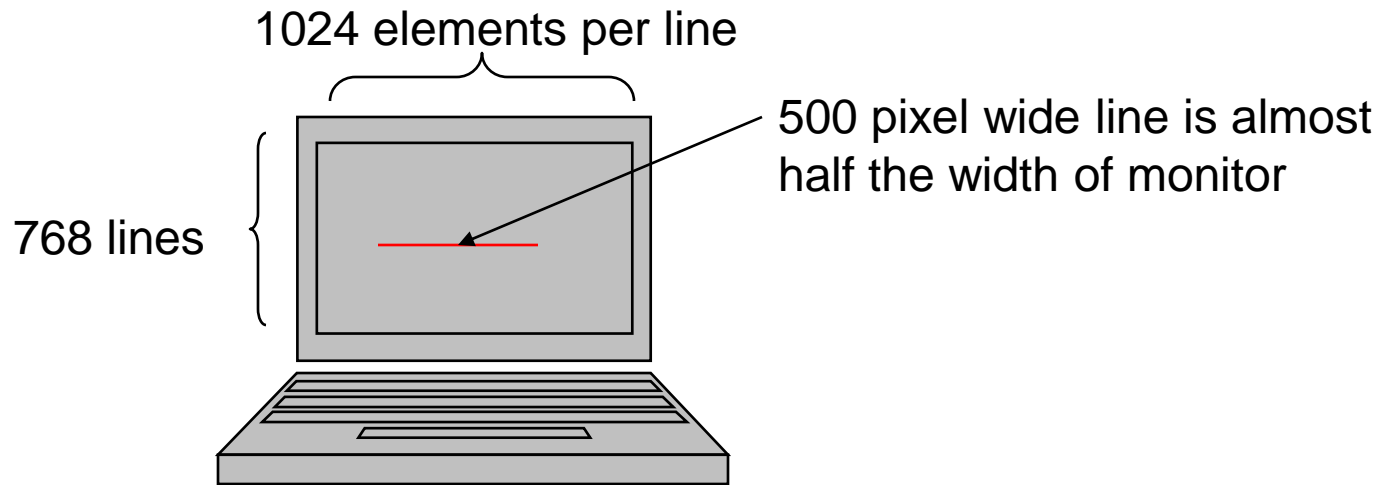
- Monitor resolution determines pixel size

1024 elements per line

768 lines

500 pixel wide line is almost half the width of monitor

Common HTML Elements

- Monitor resolution determines pixel size

1280 elements per line

500 pixel wide line is less than half the width of monitor

1024 lines

Common HTML Elements

HtmlElements.html - Mozilla

Some Common HTML Elements

Simple formatting elements

```
   Use pre (for "preformatted") to
     preserve white space and use
       monospace type.
       (But note that tags such as
still work!)
```

A horizontal *separating line* is produced using **hr**:

Other elements

See the W3C HTML 4.01 Element Index for a complete list of elements.
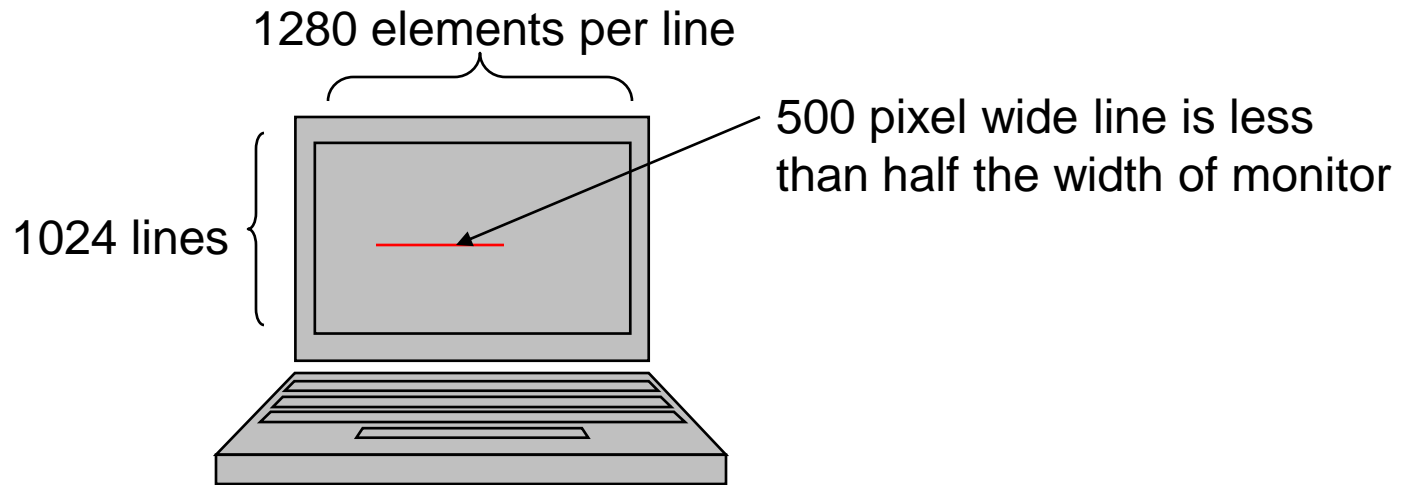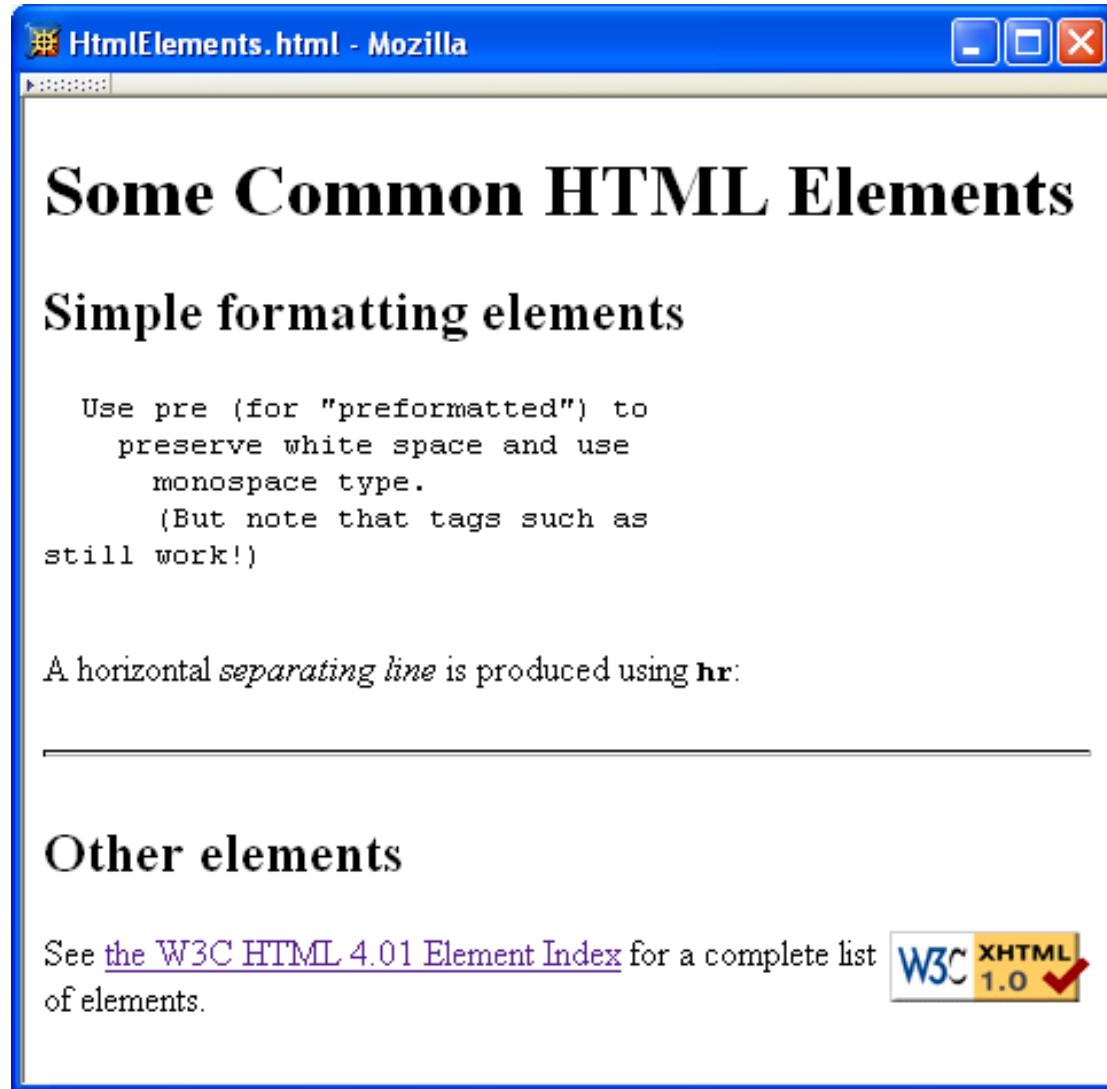
Common HTML Elements

- Hyperlinks are produced by the anchor element a

```
See
<a href="http://www.w3.org/TR/html4/index/elements.html">the
   W3C HTML 4.01 Element Index</a>
for a complete list of elements.
```

- Clicking on a hyperlink causes browser to issue GET request to URL specified in href attribute and render response in client area

- Content of anchor element is text of hyperlink (avoid leading/trailing space in content)

Common HTML Elements

- Anchors can be used as source (previous example) or destination

```
<a id="section1" name="section1"></a>
```

- The fragment portion of a URL is used to reference a destination anchor

```
<a href="http://www.example.org/PageWithAnchor.html#section1">...
```

- Browser scrolls so destination anchor is at (or near) top of client area

Common HTML Elements

- Comments are a special form of tag

```
<!-- Notice that img must nest within a "block" element,
     such as p -->
```

- Not allowed to use `--` within comment

```
        <!-- This is NOT
  ✗       -- a good comment.
           -->


  ✗     <!-- Can't end with more than two dashes! --->
```

Nesting Elements

- If one element is nested within another element, then the content of the inner element is also content of the outer element

```
<tt><strong>hr</strong></tt>
```

- XHTML requires that elements be properly nested

```
✗  <tt><strong>hr</tt></strong>
```

Nesting Elements

- Most HTML elements are either block or inline
  - Block: browser automatically generates line breaks before and after the element content
    - Ex: p
  - Inline: element content is added to the "flow"
    - Ex: `span, tt, strong, a`

Nesting Elements

- Syntactic rules of thumb:
  - Children of body must be blocks
  - Blocks can contain inline elements
  - Inline elements *cannot* contain blocks
- Specific rules for each version of (X)HTML are defined using SGML or XML (covered later)

Relative URL's

- Consider an `<img>` start tag containing attribute specification

```
src="valid-xhtml10.png"
```

- This is an example of a relative URL: it is interpreted relative to the URL of the document that contains the `img` tag
  - If document URL is
    http://localhost:8080/MultiFile.html
    then relative URL above represents absolute URL
    http://localhost:8080/valid-xhtml10.png

Relative URL's

TABLE 2.4: Absolute URL's corresponding to relative URL's when the base URL is `http://www.example.org/a/b/c.html`.

Relative URL	Absolute URL
d/e.html	http://www.example.org/a/b/d/e.html
../f.html	http://www.example.org/a/f.html
../../g.html	http://www.example.org/g.html
../h/i.html	http://www.example.org/a/h/i.html
/j.html	http://www.example.org/j.html
/k/l.html	http://www.example.org/k/l.html

Relative URL's

- Query and fragment portions of a relative URL are appended to the resulting absolute URL
  - Example: If document URL is
    http://localhost:8080/PageAnch.html
    and it contains the anchor element

    ```
    <a href="#section1">...
    ```

    then the corresponding absolute URL is
    http://localhost:8080/PageAnch.html#section1

Relative URL's

- Advantages:
  - Shorter than absolute URL's
  - Primary: can change the URL of a document (e.g., move document to a different directory or rename the server host) without needing to change URL's within the document
- Should use relative URL's whenever possible

Lists

Lists

Unordered List

```
<ul>
  <li>Bulleted list item</li>
  <li>Bulleted list item 2</li>
</ul>
```

Ordered List

```
<ol>
  <li>Numbered list item</li>
  <li>Numbered list item 2</li>
</ol>
```

Definition List

```
<dl>
  <dt>Term</dt>
  <dd>Definition of term</dd>
  <dt>Term 2</dt>
  <dd>Definition of term 2</dd>
</dl>
```
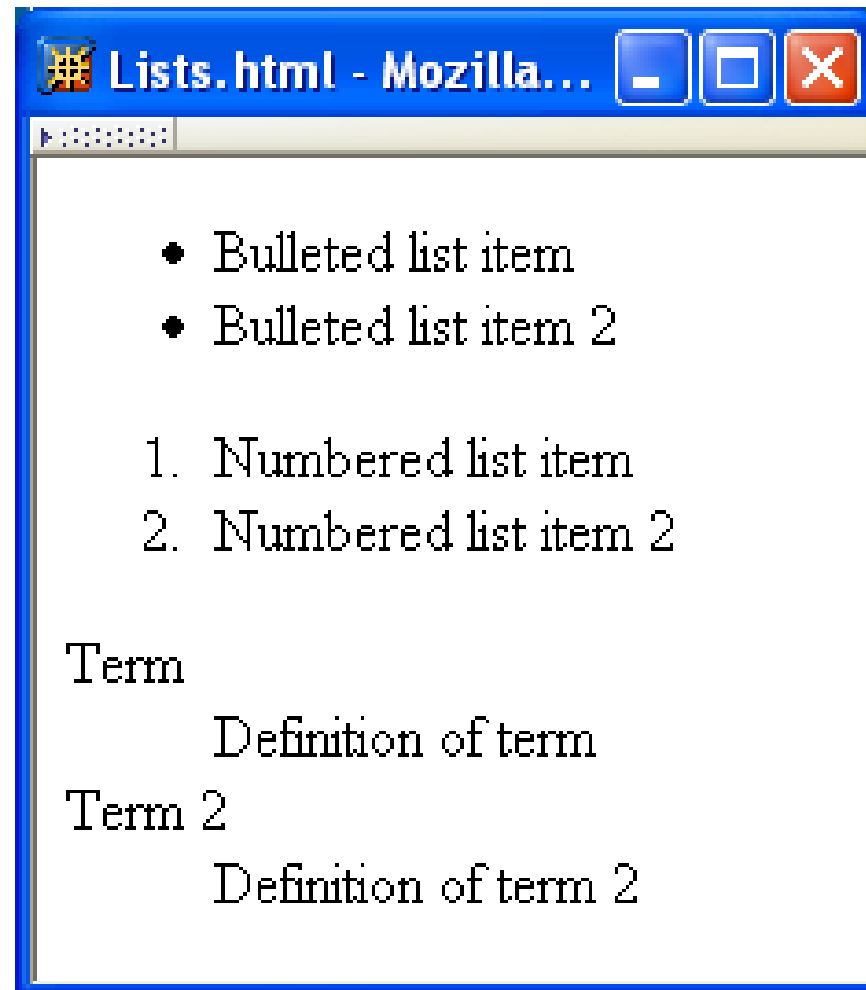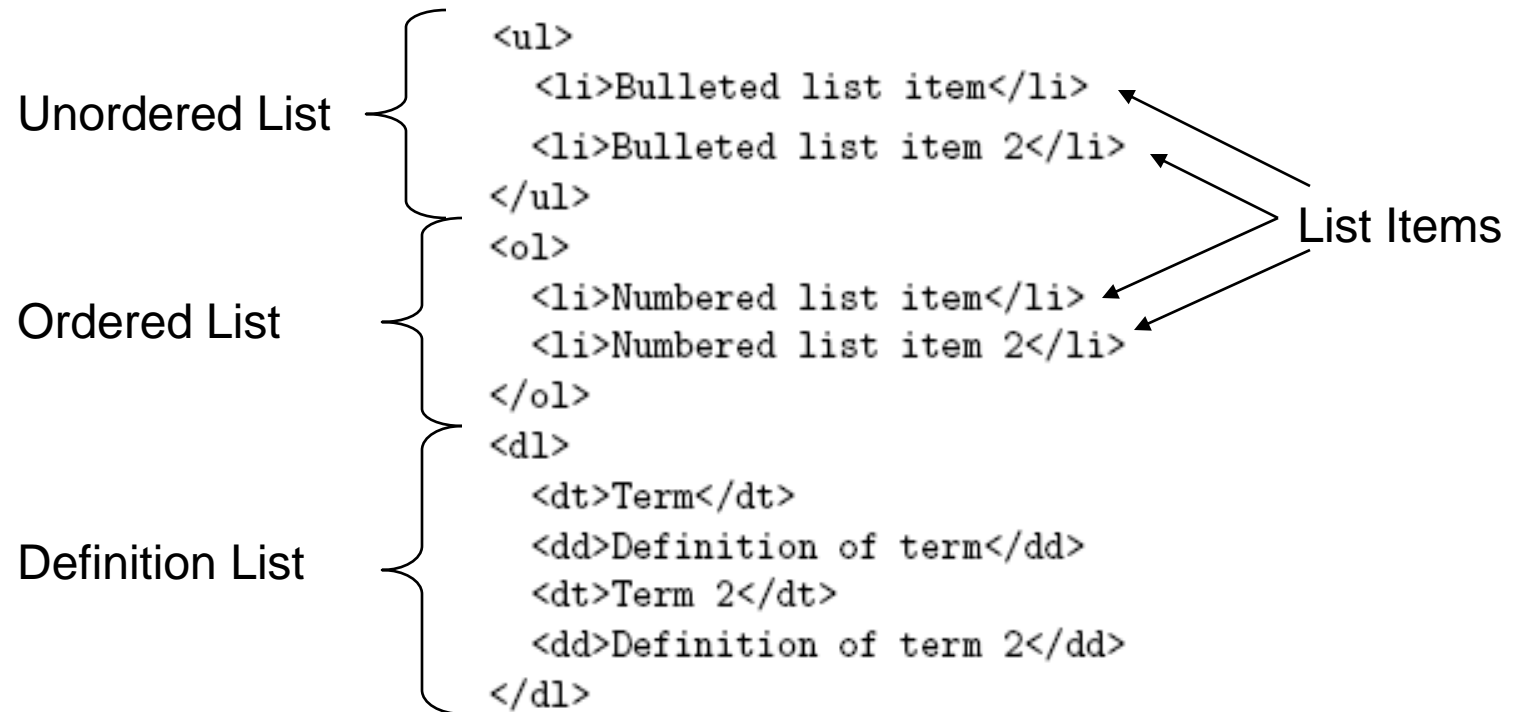
List Items

Lists


```
<ul>
   <li>Bulleted list item
      <ul>
         <li>Nested list item</li>
         <li>Nested list item 2</li>
      </ul>
   </li>
   <li>Bulleted list item 2</li>
</ul>
```

Tables

Tables

Border 5 pixels, rules 1 pixel
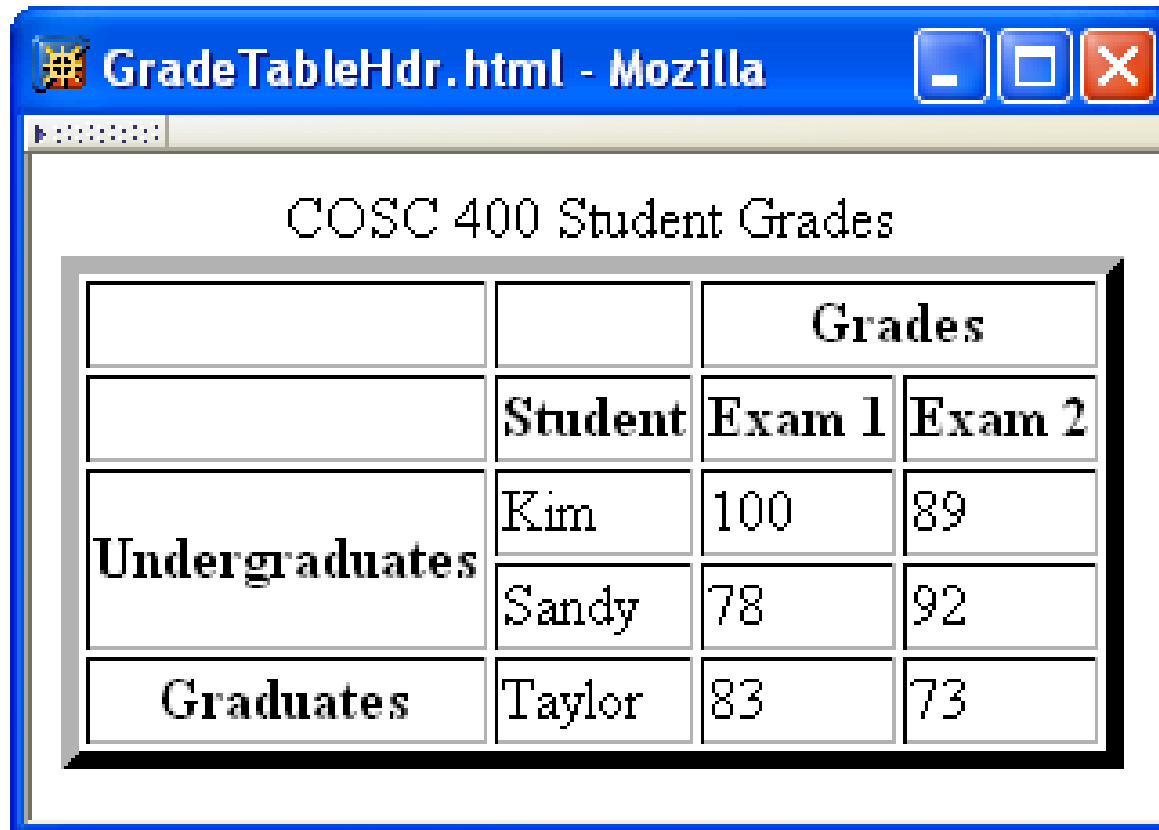
```
<table border="5">
  <tr>
    <td>Kim</td><td>100</td><td>89</td>
  </tr>
  <tr>
    <td>Sandy</td><td>78</td><td>92</td>
  </tr>
  <tr>
    <td>Taylor</td><td>83</td><td>73</td>
  </tr>
</table>
```

Table Row

Table Data

Tables

COSC 400 Student Grades

		Grades	
	Student	Exam 1	Exam 2
Undergraduates	Kim	100	89
	Sandy	78	92
Graduates	Taylor	83	73

Tables

```
<table border="5">
  <caption>
    COSC 400 Student Grades
  </caption>
  <tr>
    <td> </td><td> </td><th colspan="2">Grades</th>
  </tr>
  <tr>
    <td> </td><th>Student</th><th>Exam 1</th><th>Exam 2</th>
  </tr>
  <tr>
    <th rowspan="2">Undergraduates</th><td>Kim</td><td>100</td><td>89</td>
  </tr>
  <tr>
    <td>Sandy</td><td>78</td><td>92</td>
  </tr>
  <tr>
    <th>Graduates</th><td>Taylor</td><td>83</td><td>73</td>
  </tr>
</table>
```
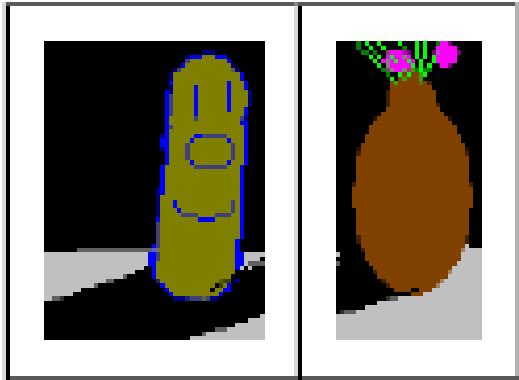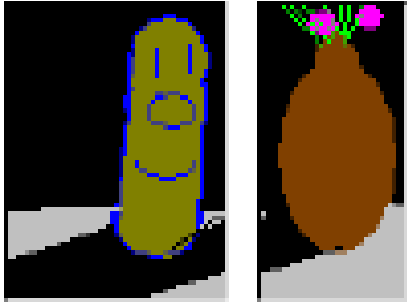
Table Header

Tables
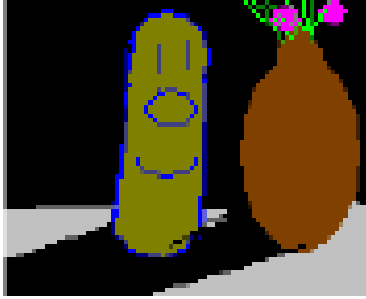

```
<table border="1" cellspacing="10" cellpadding="10">
```

Tables

Tables

cellspacing   cellpadding

Tables

cellspacing   cellpadding

Frames

Frames

```
<!DOCTYPE html
    PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Java 2 Platform SE v1.4.2</title>
  </head>
  <frameset cols="20%,80%">
    <frameset rows="1*,2*">
      <frame src="overview-frame.html"
        id="upperLeftFrame" name="upperLeftFrame"></frame>
      <frame src="allclasses-frame.html"
        id="lowerLeftFrame" name="lowerLeftFrame"></frame>
    </frameset>
    <frame src="overview-summary.html"
        id="rightFrame" name="rightFrame"></frame>
  </frameset>
</html>
```

1/3,2/3 split

Frames

- Hyperlink in one frame can load document in another:
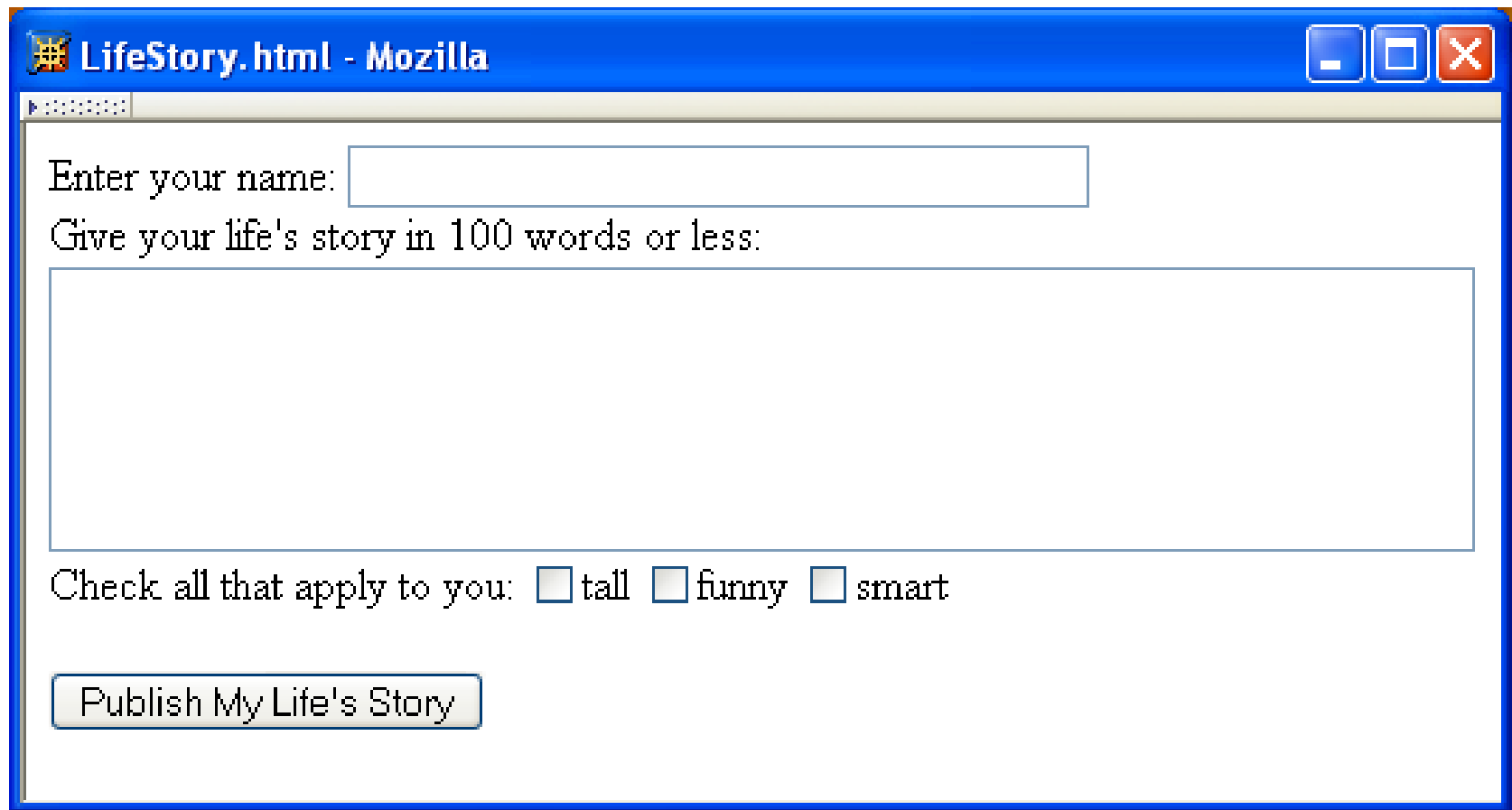
```
<a href="java/applet/package-frame.html" target="lowerLeftFrame">
```

- Value of target attribute specification is id/name of a frame

Frames

- User interface issues:
  - What happens when the page is printed?
  - What happens when the Back button is clicked?
  - How should assistive technology "read" the page?
  - How should the information be displayed on a small display?

- Recommendation: avoid frames except for applications aimed at "power users"

Forms

LifeStory.html - Mozilla

Enter your name: [_____]

Give your life's story in 100 words or less:

[   ]
[   ]
[   ]
[   ]

Check all that apply to you: ☐ tall ☐ funny ☐ smart

[Publish My Life's Story]

Forms

Each form is content of a `form` element

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

Forms

action specifies URL where form data is sent in an HTTP request

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

Forms

HTTP request method (lower case)

```
<form action="http://www.example.org" method="get">
   <div>
     <label>
       Enter your name: <input type="text" name="username" size="40" />
     </label>
     <br />
     <label>
       Give your life's story in 100 words or less:
       <br />
       <textarea name="lifestory" rows="5" cols="60"></textarea>
     </label>
     <br />
```

Forms

```
<form action="http://www.example.org" method="get">
  <div>     div is the block element analog of span (no-style block element)
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

Forms

```
<form action="http://www.example.org" method="get">
  <div>    Form control elements must be content of a block element
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

Forms

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

Text field *control* (form user-interface element)

Forms

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

Text field used for one-line inputs

77

Forms

Forms

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```
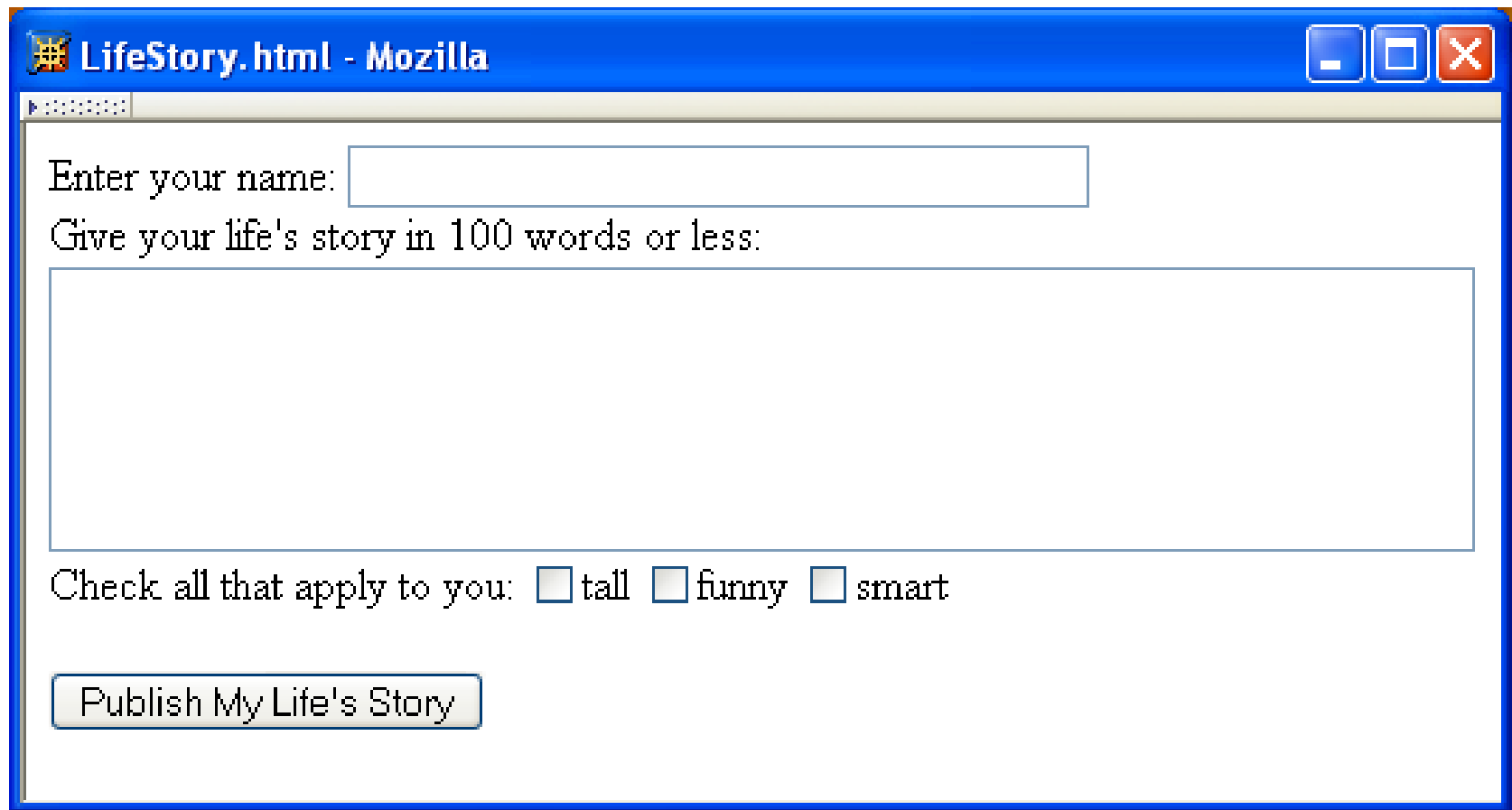
Name associated with this control's data in HTTP request

Forms

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

Width (number of characters) of text field

Forms

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

input is an empty element

Forms

```
<form action="http://www.example.org" method="get">
  <div>
    <label>    Use label to associate text with a control
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

Forms

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />    Form controls are inline elements
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

83

Forms

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

textarea control used for multi-line input

Forms

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```

Height and width in characters

Forms

```
<form action="http://www.example.org" method="get">
  <div>
    <label>
      Enter your name: <input type="text" name="username" size="40" />
    </label>
    <br />
    <label>
      Give your life's story in 100 words or less:
      <br />
      <textarea name="lifestory" rows="5" cols="60"></textarea>
    </label>
    <br />
```
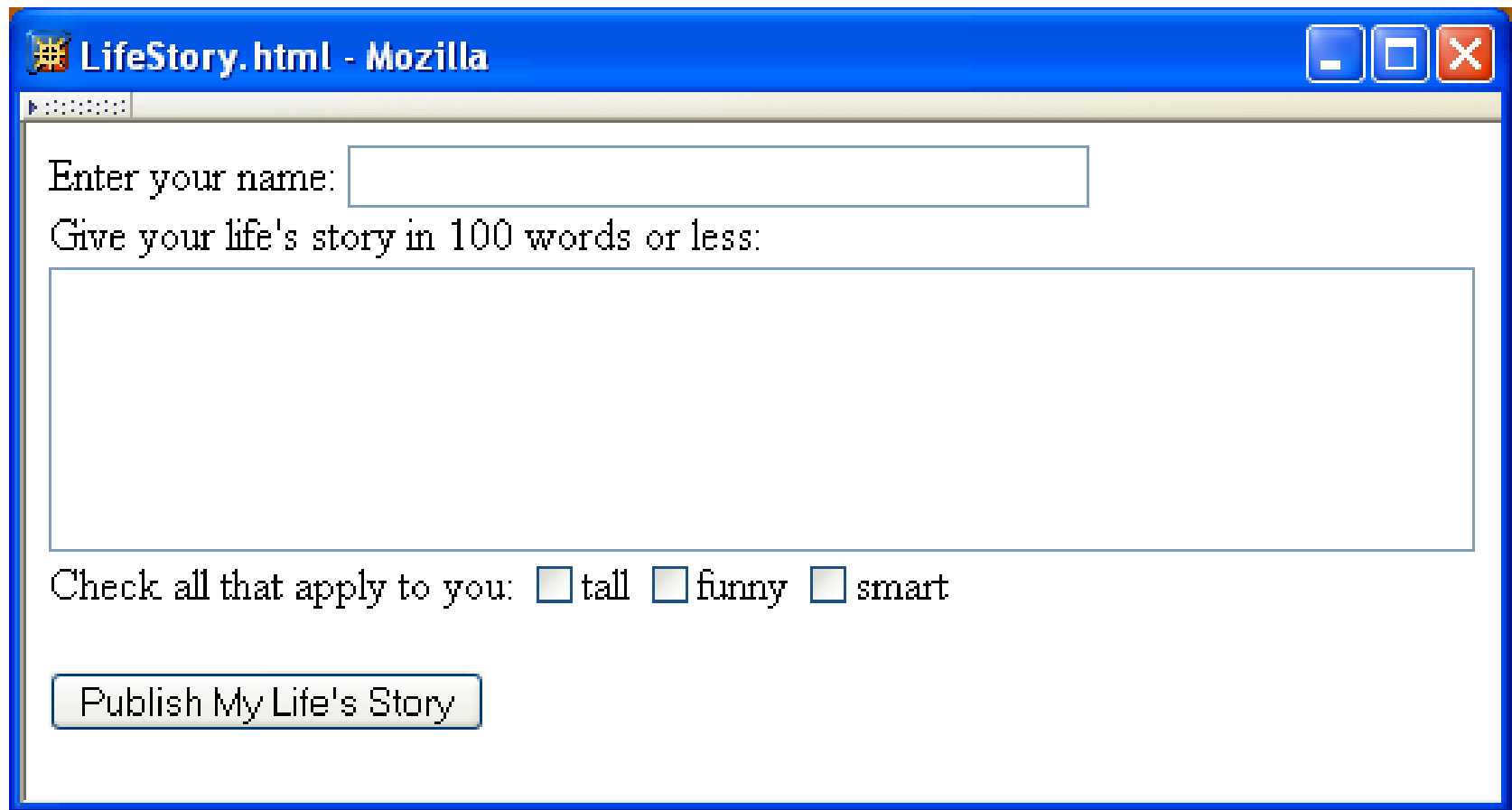
textarea is not an empty element; any content is displayed

Forms

Forms

```
Check all that apply to you:
<label>                          Checkbox control
  <input type="checkbox" name="boxgroup1" value="tall" />tall
</label>
<label>
  <input type="checkbox" name="boxgroup1" value="funny" />funny
</label>
<label>
  <input type="checkbox" name="boxgroup1" value="smart" />smart
</label>
<br /><br />
<input type="submit" name="doit" value="Publish My Life's Story" />
  </div>
</form>
```

Forms

```
Check all that apply to you:
<label>
    <input type="checkbox" name="boxgroup1" value="tall" />tall
</label>
<label>
    <input type="checkbox" name="boxgroup1" value="funny" />funny
</label>
<label>
    <input type="checkbox" name="boxgroup1" value="smart" />smart
</label>
<br /><br />
<input type="submit" name="doit" value="Publish My Life's Story" />
    </div>
</form>
```

Value sent in HTTP request if box is checked

Forms

Controls can share a common name

```
Check all that apply to you:
<label>
  <input type="checkbox" name="boxgroup1" value="tall" />tall
</label>
<label>
  <input type="checkbox" name="boxgroup1" value="funny" />funny
</label>
<label>
  <input type="checkbox" name="boxgroup1" value="smart" />smart
</label>
<br /><br />
<input type="submit" name="doit" value="Publish My Life's Story" />
  </div>
</form>
```

Forms

```
Check all that apply to you:
<label>
  <input type="checkbox" name="boxgroup1" value="tall" />tall
</label>
<label>
  <input type="checkbox" name="boxgroup1" value="funny" />funny
</label>
<label>
  <input type="checkbox" name="boxgroup1" value="smart" />smart
</label>
<br /><br />
<input type="submit" name="doit" value="Publish My Life's Story" />
</div>
</form>
```

Submit button: form data sent to action URL if button is clicked

Forms

Forms

Form data (in GET request)

Forms

```
Check all that apply to you:
<label>
  <input type="checkbox" name="boxgroup1" value="tall" />tall
</label>
<label>
  <input type="checkbox" name="boxgroup1" value="funny" />funny
</label>
<label>
  <input type="checkbox" name="boxgroup1" value="smart" />smart
</label>
<br /><br />
<input type="submit" name="doit" value="Publish My Life's Story" />
  </div>
</form>
```

Displayed on button and sent to server if button clicked

Forms

Radio buttons: at most one can be selected at a time.

IncomeRadio.html - Mozilla

Your annual income is (select one):
- ◯ Less than $10,000
- ◉ Between $10,000 and $50,000
- ◯ Over $50,000

Forms

```
Your annual income is (select one):<br />
<label>                    Radio button control
  <input type="radio" name="radgroup1" value="0-10" />
    Less than $10,000
</label><br />
<label>
  <input type="radio" name="radgroup1" value="10-50"
          checked="checked" />
    Between $10,000 and $50,000
</label><br />
<label>
  <input type="radio" name="radgroup1" value="&gt;50" />
    Over $50,000
</label>
```

Forms

```
Your annual income is (select one):<br />
<label>
   <input type="radio" name="radgroup1" value="0-10" />
      Less than $10,000
</label><br />
<label>
   <input type="radio" name="radgroup1" value="10-50"
          checked="checked" />
      Between $10,000 and $50,000
</label><br />
<label>
   <input type="radio" name="radgroup1" value="&gt;50" />
      Over $50,000
</label>
```

All radio buttons with the same name form a *button set*

Forms

```
Your annual income is (select one):<br />
<label>
  <input type="radio" name="radgroup1" value="0-10" />
    Less than $10,000
</label><br />
<label>
  <input type="radio" name="radgroup1" value="10-50"
         checked="checked" />
    Between $10,000 and $50,000
</label><br />
<label>
  <input type="radio" name="radgroup1" value="&gt;50" />
    Over $50,000
</label>
```

Only one button of a set can be selected at a time

Forms

```
Your annual income is (select one):<br />
<label>
  <input type="radio" name="radgroup1" value="0-10" />
    Less than $10,000
</label><br />
<label>
  <input type="radio" name="radgroup1" value="10-50"
    checked="checked" />
    Between $10,000 and $50,000
</label><br />
<label>
  <input type="radio" name="radgroup1" value="&gt;50" />
    Over $50,000
</label>
```

This button is initially selected (`checked` attribute also applies to check boxes)

Forms

```
Your annual income is (select one):<br />
<label>
  <input type="radio" name="radgroup1" value="0-10" />
    Less than $10,000
</label><br />
<label>
  <input type="radio" name="radgroup1" value="10-50"
        checked="checked" />
    Between $10,000 and $50,000
</label><br />
<label>
  <input type="radio" name="radgroup1" value="&gt;50" />
    Over $50,000
</label>
```

Boolean attribute: default false, set true by specifying name as value

Forms

```
Your annual income is (select one):<br />
<label>
  <input type="radio" name="radgroup1" value="0-10" />
    Less than $10,000
</label><br />
<label>
  <input type="radio" name="radgroup1" value="10-50"
         checked="checked" />
    Between $10,000 and $50,000
</label><br />
<label>
  <input type="radio" name="radgroup1" value="&gt;50" />
    Over $50,000
</label>
```

Represents string: >50

Forms

Forms

```
Your annual income is (select one):
<select name="income">        Menu control; name given once
  <option value="0-10">Less than $10,000</option>
  <option value="10-50" selected="selected">
    Between $10,000 and $50,000
  </option>
  <option value="&gt;50">Over $50,000</option>
</select>
```

Forms

```
Your annual income is (select one):
<select name="income">
  <option value="0-10">Less than $10,000</option>
  <option value="10-50" selected="selected">
    Between $10,000 and $50,000
  </option>
  <option value="&gt;50">Over $50,000</option>
</select>
```

Each menu item has its own value

Forms

```
Your annual income is (select one):
<select name="income">
  <option value="0-10">Less than $10,000</option>
  <option value="10-50" selected="selected">
    Between $10,000 and $50,000
  </option>
  <option value="&gt;50">Over $50,000</option>
</select>
```
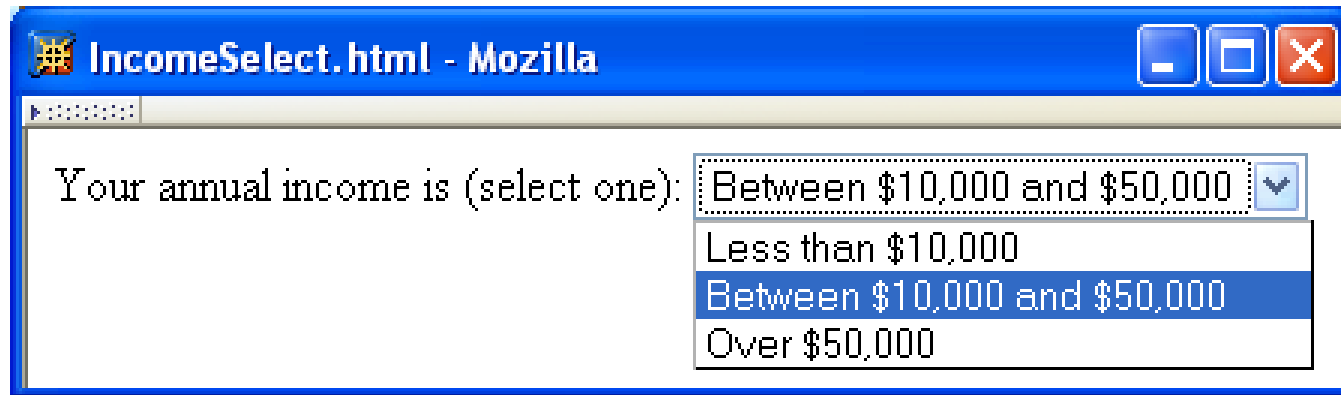
Item initially displayed in menu control

Forms

- Other form controls:
  - Fieldset (grouping)
  - Password
  - Clickable image
  - Non-submit buttons
  - Hidden (embed data)
  - File upload
  - Hierarchical menus

Forms

TABLE 2.5: HTML 4.01/XHTML 1.0 non-deprecated form controls.

Element	type Attribute	Control
input	text	Text input
input	password	Password input
input	checkbox	Checkbox
input	radio	Radio button
input	submit	Submit button
input	image	Graphical submit button
input	reset	Reset button (form clear)
input	button	Push button (for use with scripts)
input	hidden	Non-displayed control (stores server-supplied information)
input	file	File select
button	submit	Submit button with content (not an empty element)
button	reset	Cancel button with content (not an empty element)
button	button	Button with content but no predefined action
select	N/A	Menu
option	N/A	Menu item
optgroup	N/A	Heading in a hierarchical menu
textarea	N/A	Multi-line text input
label	N/A	Associate label with control(s)
fieldset	N/A	Groups controls
legend	N/A	Add caption to a fieldset

XML DTD

- Recall that XML is used to define the syntax of XHTML
- Set of XML files that define a language are known as the document type definition (DTD)
- DTD primarily consists of declarations:
  - Element type: name and content of elements
  - Attribute list: attributes of an element
  - Entity: define meaning of, *e.g.*, >

XML Element Type Declaration

```
<!ELEMENT html (head, body)>
```

Element type name

XML Element Type Declaration

```
<!ELEMENT html (head, body)>
```

Element type *content specification* (or *content model*)

XML Element Type Declaration

```
<!ELEMENT html (head, body)>
```

Element type *content specification* (or *content model*)

TABLE 2.6: Basic XML content specifications.

Specification Type	Syntax	Content Allowed
Empty	EMPTY	None
Arbitrary	ANY	Any content (no restrictions)
Sequence	(elt1, elt2, ...)	Sequence of elements that must appear in order specified
Choice	(elt1 | elt2 | ...)	Exactly one of the specified elements must appear
Character data	(#PCDATA)	Arbitrary character data, but no elements
Mixed	(#PCDATA | elt1 | elt2 |...)*	Any mixture of character data and the specified elements in any order

XML Element Type Declaration

`<!ELEMENT html (head, body)>`

Element type *content specification* (or *content model*)

TABLE 2.6: Basic XML content specifications.

Specification Type	Syntax	Content Allowed
Empty	EMPTY	None
Arbitrary	ANY	Any content (no restrictions)
Sequence	(elt1, elt2, ...)	Sequence of elements that must appear in order specified
Choice	(elt1 | elt2 | ...)	Exactly one of the specified elements must appear
Character data	(#PCDATA)	Arbitrary character data, but no elements
Mixed	(#PCDATA | elt1 | elt2 |...)*	Any mixture of character data and the specified elements in any order

XML Element Type Declaration

```
<!ELEMENT br EMPTY>
```

Element type *content specification* (or *content model*)

TABLE 2.6: Basic XML content specifications.

Specification Type	Syntax	Content Allowed
Empty	EMPTY	None
Arbitrary	ANY	Any content (no restrictions)
Sequence	(elt1, elt2, ...)	Sequence of elements that must appear in order specified
Choice	(elt1 | elt2 | ...)	Exactly one of the specified elements must appear
Character data	(#PCDATA)	Arbitrary character data, but no elements
Mixed	(#PCDATA | elt1 | elt2 |...)*	Any mixture of character data and the specified elements in any order

XML Element Type Declaration

```
<!ELEMENT br EMPTY>
```

Element type *content specification* (or *content model*)

TABLE 2.6: Basic XML content specifications.

Specification Type	Syntax	Content Allowed
Empty	EMPTY	None
Arbitrary	ANY	Any content (no restrictions)
Sequence	(elt1, elt2, ...)	Sequence of elements that must appear in order specified
Choice	(elt1 | elt2 | ...)	Exactly one of the specified elements must appear
Character data	(#PCDATA)	Arbitrary character data, but no elements
Mixed	(#PCDATA | elt1 | elt2 |...)*	Any mixture of character data and the specified elements in any order

XML Element Type Declaration

`<!ELEMENT select (optgroup|option)+>`

Element type *content specification* (or *content model*)

TABLE 2.6: Basic XML content specifications.

Specification Type	Syntax	Content Allowed
Empty	EMPTY	None
Arbitrary	ANY	Any content (no restrictions)
Sequence	(elt1, elt2, ...)	Sequence of elements that must appear in order specified
Choice	(elt1 | elt2 | ...)	Exactly one of the specified elements must appear
Character data	(#PCDATA)	Arbitrary character data, but no elements
Mixed	(#PCDATA | elt1 | elt2 |...)*	Any mixture of character data and the specified elements in any order

115

XML Element Type Declaration

`<!ELEMENT select (optgroup|option)+>`

Element type *content specification* (or *content model*)

TABLE 2.6: Basic XML content specifications.

Specification Type	Syntax	Content Allowed
Empty	EMPTY	None
Arbitrary	ANY	Any content (no restrictions)
Sequence	(elt1, elt2, ...)	Sequence of elements that must appear in order specified
Choice	(elt1 | elt2 | ...)	Exactly one of the specified elements must appear
Character data	(#PCDATA)	Arbitrary character data, but no elements
Mixed	(#PCDATA | elt1 | elt2 |...)*	Any mixture of character data and the specified elements in any order

116

XML Element Type Declaration

<!ELEMENT textarea (#PCDATA)>

Element type *content specification* (or *content model*)

TABLE 2.6: Basic XML content specifications.

Specification Type	Syntax	Content Allowed
Empty	EMPTY	None
Arbitrary	ANY	Any content (no restrictions)
Sequence	(elt1, elt2, ...)	Sequence of elements that must appear in order specified
Choice	(elt1 | elt2 | ...)	Exactly one of the specified elements must appear
Character data	(#PCDATA)	Arbitrary character data, but no elements
Mixed	(#PCDATA | elt1 | elt2 |...)*	Any mixture of character data and the specified elements in any order

XML Element Type Declaration

<!ELEMENT textarea (#PCDATA)>

Element type *content specification* (or *content model*)

TABLE 2.6: Basic XML content specifications.

Specification Type	Syntax	Content Allowed
Empty	EMPTY	None
Arbitrary	ANY	Any content (no restrictions)
Sequence	(elt1, elt2, ...)	Sequence of elements that must appear in order specified
Choice	(elt1 | elt2 | ...)	Exactly one of the specified elements must appear
Character data	(#PCDATA)	Arbitrary character data, but no elements
Mixed	(#PCDATA | elt1 | elt2 |...)*	Any mixture of character data and the specified elements in any order

XML Element Type Declaration

```
<!ELEMENT select (optgroup|option)+>
```

Element type *content specification* (or *content model*)

XML Element Type Declaration

```
<!ELEMENT select (optgroup|option)+>
```

Element type *content specification* (or *content model*)

TABLE 2.7: XML content specification iterator characters.

Character	Meaning
?	Sequence/choice is optional (appears zero or one times)
*	Sequence/choice may be repeated an arbitrary number of times, including none
+	Sequence/choice may appear one or more times

XML Element Type Declaration

```
<!ELEMENT select (optgroup|option)+>
```

Element type *content specification* (or *content model*)

TABLE 2.7: XML content specification iterator characters.

Character	Meaning
?	Sequence/choice is optional (appears zero or one times)
*	Sequence/choice may be repeated an arbitrary number of times, including none
+	Sequence/choice may appear one or more times

XML Element Type Declaration

```
<!ELEMENT table
     (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:

XML Element Type Declaration

```
<!ELEMENT table
      (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption`

XML Element Type Declaration

```
<!ELEMENT table
    (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by

XML Element Type Declaration

```
<!ELEMENT table
    (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements

XML Element Type Declaration

```
<!ELEMENT table
    (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements or

XML Element Type Declaration

```
<!ELEMENT table
    (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements or any number of `colgroup` elements

XML Element Type Declaration

```
<!ELEMENT table
    (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements or any number of `colgroup` elements then

128

XML Element Type Declaration

```
<!ELEMENT table
    (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements or any number of `colgroup` elements then
  - Optional header

XML Element Type Declaration

```
<!ELEMENT table
     (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements or any number of `colgroup` elements then
  - Optional header followed by

XML Element Type Declaration

```
<!ELEMENT table
    (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements or any number of `colgroup` elements then
  - Optional header followed by optional footer

XML Element Type Declaration

```
<!ELEMENT table
    (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements or any number of `colgroup` elements then
  - Optional header followed by optional footer then

XML Element Type Declaration

```
<!ELEMENT table
    (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements or any number of `colgroup` elements then
  - Optional header followed by optional footer then
  - One or more `tbody` elements

XML Element Type Declaration

```
<!ELEMENT table
    (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements or any number of `colgroup` elements then
  - Optional header followed by optional footer then
  - One or more `tbody` elements or

XML Element Type Declaration

```
<!ELEMENT table
     (caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|(tr+))>
```

- Child elements of `table` are:
  - Optional `caption` followed by
  - Any number of `col` elements or any number of `colgroup` elements then
  - Optional header followed by optional footer then
  - One or more `tbody` elements or one or more `tr` elements

XML Attribute List Declaration

Element type name

```
<!ATTLIST html
   lang          NMTOKEN    #IMPLIED
   xml:lang      NMTOKEN    #IMPLIED
   dir           (ltr|rtl)  #IMPLIED
   id            ID         #IMPLIED
   xmlns         CDATA      #FIXED 'http://www.w3.org/1999/xhtml'>
```

XML Attribute List Declaration

```
<!ATTLIST html
    lang        NMTOKEN     #IMPLIED
    xml:lang    NMTOKEN     #IMPLIED
    dir         (ltr|rtl)   #IMPLIED
    id          ID          #IMPLIED
    xmlns       CDATA       #FIXED 'http://www.w3.org/1999/xhtml'>
```

Recognized
attribute names

XML Attribute List Declaration

```
<!ATTLIST html
    lang           NMTOKEN     #IMPLIED
    xml:lang       NMTOKEN     #IMPLIED
    dir            (ltr|rtl)   #IMPLIED
    id             ID          #IMPLIED
    xmlns          CDATA       #FIXED 'http://www.w3.org/1999/xhtml'>
```

Attribute types
(data types allowed as attribute values)

138

XML Attribute List Declaration

ASCII characters: letter, digit, or . - _ :

```
<!ATTLIST html
    lang            NMTOKEN     #IMPLIED
    xml:lang        NMTOKEN     #IMPLIED
    dir             (ltr|rtl)   #IMPLIED
    id              ID          #IMPLIED
    xmlns           CDATA       #FIXED 'http://www.w3.org/1999/xhtml'>
```

XML Attribute List Declaration

Attribute value must be ltr or rtl

```
<!ATTLIST html
   lang          NMTOKEN    #IMPLIED
   xml:lang      NMTOKEN    #IMPLIED
   dir           (ltr|rtl)  #IMPLIED
   id            ID         #IMPLIED
   xmlns         CDATA      #FIXED 'http://www.w3.org/1999/xhtml'>
```

XML Attribute List Declaration

```
<!ATTLIST html
   lang          NMTOKEN    #IMPLIED
   xml:lang      NMTOKEN    #IMPLIED
   dir           (ltr|rtl)  #IMPLIED
   id            ID         #IMPLIED
   xmlns         CDATA      #FIXED 'http://www.w3.org/1999/xhtml'>
```

Like NMTOKEN but must begin with letter or _ :
Attribute value must be unique

141

XML Attribute List Declaration

```
<!ATTLIST html
    lang        NMTOKEN     #IMPLIED
    xml:lang    NMTOKEN     #IMPLIED
    dir         (ltr|rtl)   #IMPLIED
    id          ID          #IMPLIED
    xmlns       CDATA       #FIXED 'http://www.w3.org/1999/xhtml'>
```

Any character except XML special characters < and &
or the quote character enclosing the attribute value

XML Attribute List Declaration

TABLE 2.8: Key attribute types used in XHTML 1.0 Strict DTD.

Attribute type	Syntax	Usage
Name token	NMTOKEN	Name (word)
Enumerated	(string1 | string2 | ...)	List of all possible attribute values
Identifier	ID	Type for id attribute
Identifier reference	IDREF	Reference to an id attribute value
Identifier reference list	IDREFS	List of references to id attribute values
Character data	CDATA	Arbitrary character data (except < and &)

XML Attribute List Declaration

```
<!ATTLIST html
    lang          NMTOKEN    #IMPLIED
    xml:lang      NMTOKEN    #IMPLIED
    dir           (ltr|rtl)  #IMPLIED
    id            ID         #IMPLIED
    xmlns         CDATA      #FIXED 'http://www.w3.org/1999/xhtml'>
```

Attribute default declarations

XML Attribute List Declaration

TABLE 2.9: XML attribute default-value declarations.

Default type	Syntax
No default value provided by DTD, attribute optional	#IMPLIED
Default provided by DTD, may not be changed	#FIXED followed by any valid value (quoted)
Default provided by DTD, may be overridden by user	Any valid value (quoted)
No default value provided by DTD, attribute required	#REQUIRED

XML Entity Declaration

- Entity declaration is essentially a macro
- Two types of entity:
  - General: referenced from HTML document using &

```
<!ENTITY gt        "&#62;">
```

Entity name

XML Entity Declaration

- Entity declaration is essentially a macro

- Two types of entity:

  – General: referenced from HTML document using &

```
<!ENTITY gt          "&#62;">
```

Replacement text;
recursively replaced if it is a reference

XML Entity Declaration

- Entity declaration is essentially a macro

- Two types of entity:
  - General: referenced from HTML document using &

  ```
  <!ENTITY gt        "&#62;">
  ```

  - Parameter: reference from DTD using %

  ```
  <!ENTITY % LanguageCode "NMTOKEN">

  <!ATTLIST html
    lang          NMTOKEN    #IMPLIED
    xml:lang      %LanguageCode; #IMPLIED
  ```

XML Entity Declaration

- Entity declaration is essentially a macro
- Two types of entity:
  - General: referenced from HTML document using &

    ```
    <!ENTITY gt        "&#62;">
    ```

  - Parameter: reference from DTD using %

    ```
    <!ENTITY % LanguageCode "NMTOKEN">

    <!ATTLIST html
      lang          NMTOKEN    #IMPLIED
      xml:lang      %LanguageCode;  #IMPLIED
    ```

DTD Files

```
<!DOCTYPE html
        PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

System Identifier: URL for primary DTD document

- DTD document contains element type, attribute list, and entity declarations

- May also contain declaration of external entities: identifiers for secondary DTD documents

150

DTD Files

External entity name

```
<!ENTITY % HTMLlat1 PUBLIC
    "-//W3C//ENTITIES Latin 1 for XHTML//EN"
    "xhtml-lat1.ent">
%HTMLlat1;
```

DTD Files

```
<!ENTITY % HTMLlat1 PUBLIC
    "-//W3C//ENTITIES Latin 1 for XHTML//EN"
    "xhtml-lat1.ent">
%HTMLlat1;
```
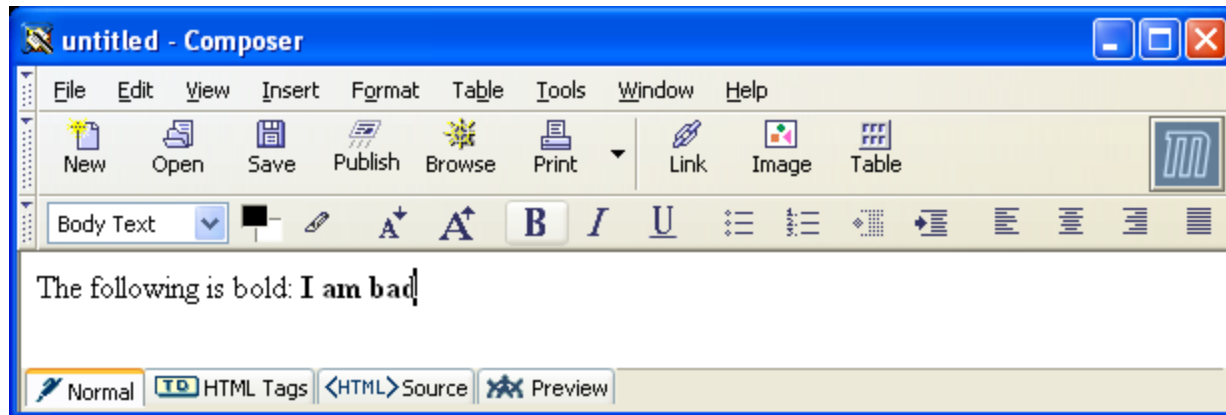
System identifier (relative URL)

DTD Files

```
<!ENTITY % HTMLlat1 PUBLIC
    "-//W3C//ENTITIES Latin 1 for XHTML//EN"
    "xhtml-lat1.ent">
%HTMLlat1;
```

Entity reference; imports content (entity declarations, called *entity set*) of external entity at this point in the primary DTD

HTML Creation Tools

- ## Mozilla Composer

- ## Microsoft FrontPage
- ## Macromedia Dreamweaver
- ## Etc.

Case Study

Case Study

```
<!DOCTYPE html
        PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<!-- login.html -->
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Login to My Own Blog!</title>
  </head>
  <body>
    <h1>
      Log in to My Own Blog to begin blogging!
    </h1>
    <form action="Login" method="post">
      <table border="0">
        <tbody>
          <tr>
            <td>User Name:</td>
            <td><input type="text" name="username" /></td>
          </tr>
```

Borderless table used to lay out form

Case Study

```
<tr>
    <td>Password:</td>
    <td><input type="password" name="pwd" /></td>
</tr>
<tr>
    <td> </td>
    <td><input type="submit" name="submit"
               value="Log in" /></td>
</tr>
</tbody>
</table>
</form>
</body>
</html>
```

Special text field for passwords

Case Study

Use ref. to get <

Fix this later with "style"

158

Case Study

Case Study

```
<div>
  <!-- Banner image -->
  <img src="banner.gif" width="438" height="120"
      alt='"My Own Blog!" Banner' />
</div>
<table border="0">
  <tr>
    <!-- Blog entries -->
    <td style="vertical-align:top">
      ...
    </td>
    <!-- Side information -->
    <td style="vertical-align:top">
      ...
    </td>
  </tr>
</table>
```

Banner

Blog entries

Side information

Table used for side-by-side layout

Case Study:
Blog Entry

```
AUGUST 9, 2005, 5:04 PM EDT
<h1>I'm hungry</h1>
<p>
  Strange.  I seem to get hungry about the same time
  every day.  Maybe it's something in the water.
</p>
<hr />
```

Case Study:
Side Information

```
<ul>
    <li><a href="index.html">Home</a></li>
    <li>Archives
        <ul>
            <li><a href="index.html?month=8&amp;year=2005"
                >August 2005</a></li>
            <li><a href="index.html?month=7&amp;year=2005"
                >July 2005</a></li>
            <li><a href="index.html?month=5&amp;year=2005"
                >May 2005</a></li>
        </ul>
    </li>
</ul>
```

Represent & in attribute value

Keep month and year together

References

- XHTML 1.0
  - Semantics: HTML 4.01
    - Index of all elements
  - Syntax restrictions: XHTML 1.0
  - Annotated DTD
- Relative URL
- Accessibility guidelines
- XML 1.0