Topic 2 XHTML
CITS3403 Internet Technologies

Reference: Sebesta Chapter 2

http://undergraduate.csse.uwa.edu.au/units/CITS3403/sebesta-7e/w7code2/
Origins and Evolutions of HTML

- HTML was defined with SGML
- Original intent of HTML: General layout 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
    - Just 4.01 defined using XML, instead of SGML
  - XHTML 1.1 – 2001
    - Modularized 1.0, and drops frames
    - Although the value of the consistent and coherent syntax rules of XHTML were widely recognized and accepted, its draconian error handling was not, so XHTML documents are served as HTML
Origins and Evolutions of HTML (cont.)

• W3C worked on XHTML 2.0; WHAT worked on a new version of HTML

• In 2009, XHTML 2.0 was stopped; W3C took over development of HTML5

• Reasons to use XHTML syntax rules:
  – HTML has lax syntax rules, leading to sloppy and sometimes to ambiguous documents
  – XHTML syntax is much more strict, leading to clean and clear documents in a standard form
  – HTML processors do not even enforce the few syntax rule that do exist in HTML
  – The syntactic correctness of XHTML documents can be validated

• In this unit, HTML5 is used, but with XHTML syntax rules
XHTML References

• We will give just a brief overview of common aspects of XHTML. Many references can be found in texts and on-line

• For the official standards see also:
  – Normative definition of XHTML 1.0
    • http://www.w3.org/TR/xhtml1/
  – XHTML 1.0 Strict DTD
    • http://www.w3.org/TR/xhtml1/dtds.html#a_dtd_XHTML-1.0-Strict
  – Differences of XHTML1 from HTML4
    • http://www.w3.org/TR/xhtml1/#diffs
  – XHTML Modularisation 1.1
    • http://www.w3.org/TR/xhtml-modularization/
  – XHTML 2.0 Working Draft
    • http://www.w3.org/TR/2006/WD-xhtml2-20060726/
XHTML Basic Syntax

- Elements 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
  - 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*
XHTML Syntax

• If a tag has *attributes*, they appear between its name and the right bracket of the opening tag

  – eg. `<table border="1" cellpadding="1">...</table>`

• Comments:
  `<!-- ignore stuff here... -->`

• Browsers ignore comments, unrecognizable tags, line breaks, multiple spaces, and tabs

• Tags are suggestions to the browser, can be ignored (even if they are recognized by the browser)
Logical Markup versus Display Markup

• One of the ideas (ideals?) behind the web and (other places such as great typesetting languages like LaTeX) is the separation between *what sort of information* it is, and *how it should be displayed*
  
  – allows the same source information to be easily communicated in multiple environments
  
  – eg “*this is a heading*” or “*this should be emphasised*” should not determine how it should be displayed

  • computer browser
  
  • mobile phone
  
  • interface for partially sighted (eg. read out)
Logical Markup versus Display Markup

- HTML didn’t work out
  - included tags that specify typeface - font styles and sizes
    - eg. `<b>`, `<i>`, `<p align="right">`
  - page designers
    - less concerned about ideals and more about standardising their pages
    - demanded features which browser makers included
    - made their way into HTML (or effective HTML) by the “back door”
  - result
    - bloated
    - loss of standard
    - “features” that worked differently (if at all) on different browsers
Logical Markup versus Display Markup

• XHTML tries to return as far as possible to this ideal
  – use Cascading Style Sheets (CSS) to define display properties
    • (see next topic...)
  – use XHTML for logical markup
  – eg.
    • `<em>` - this text is special, emphasise it in some way (usually italics)
    • `<strong>` - even more emphasised (usually bold)
    • `<code>` - computer code (usually uses monospaced font)
    • `<sup>` - superscript (smaller and higher)
    • `<sub>` - subscript (smaller and lower)
    • `<h1>` - highest level heading (usually very large, maybe centred)
    • `<blockquote>` - a quote (typically indented, maybe italics)
    • etc
XHTML Document Structure

• Every XHTML document must begin with xml declaration, eg:

```xml
<?xml version = "1.0" encoding = "utf-8"?>
```

• Then SGML DOCTYPE command, eg:

```xml
<!DOCTYPE html PUBLIC "-//w3c//DTD XHTML 1.1//EN"

http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd>
```

• dtd – Document Type Definition
  – specification with which document complies
  – can be used for validation (check it out)
XHTML Document Structure

• The whole document must have `<html>` as its root

html 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 used to give the document a title, which is normally displayed in the browser’s window title bar (at the top of the display)

• Prior to XHTML 1.1, a document could have either a body or a frameset
XHTML Document Structure

- Document is a *tree* of elements
- Visible elements are on `<body>` branch
  - eg text is most commonly in paragraph `<p>` elements
  - (note indentation matches tree depth...)

```xml
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//w3c//DTD XHTML 1.1//EN"
 http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd>
<html xmlns = "http://www.w3.org/1999/xhtml">
<head>
  <title> Our first document </title>
</head>
<body>
  <p>
    Greetings from your Webmaster!
  </p>
</body>
</html>
```
Basic Text Markup

• **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

```html
<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)

Aidan's Airplanes (h1)
The best in used airplanes (h2)
"We've got them by the hangarful" (h3)
We're the guys to see for a good used airplane (h4)
We offer great prices on great planes (h5)
No returns, no guarantees, no refunds, all sales are final! (h6)
# Character Entities

<table>
<thead>
<tr>
<th>Character</th>
<th>Entity</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;</td>
<td>&amp;</td>
<td>Ampersand</td>
</tr>
<tr>
<td>&lt;</td>
<td>&lt;</td>
<td>Less than</td>
</tr>
<tr>
<td>&gt;</td>
<td>&gt;</td>
<td>Greater than</td>
</tr>
<tr>
<td>&quot;</td>
<td>&quot;</td>
<td>Double quote</td>
</tr>
<tr>
<td>'</td>
<td>'</td>
<td>Single quote (apostrophe)</td>
</tr>
<tr>
<td>1/4</td>
<td>¼</td>
<td>One quarter</td>
</tr>
<tr>
<td>1/2</td>
<td>½</td>
<td>One half</td>
</tr>
<tr>
<td>3/4</td>
<td>¾</td>
<td>Three quarters</td>
</tr>
<tr>
<td>°</td>
<td>°</td>
<td>Degree</td>
</tr>
<tr>
<td>(space)</td>
<td> </td>
<td>Nonbreaking space</td>
</tr>
</tbody>
</table>
XHTML Document Structure

• Document is a *tree* of elements

• Visible elements are on <body> branch
  – eg text is most commonly in paragraph <p> elements
  – (note indentation matches tree depth...)

```xml
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//w3c//DTD XHTML 1.1//EN"
  http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd>
<html xmlns = "http://www.w3.org/1999/xhtml"/>
  <head>
    <title> Our first document </title>
  </head>
  <body>
    <p>
      Greetings from your Webmaster!
    </p>
  </body>
</html>
```
The HTML5 Way

• Document is a tree of elements

• No need to have \texttt{<html>}, \texttt{<head>}, and even \texttt{<body>}
  
  – No need to close \texttt{<p>} elements
  
  – This is valid HTML5 code according to

\begin{verbatim}
<!DOCTYPE html>
<html lang = "en">
<head>
  <meta charset="utf-8"/>
  <title> Our first document </title>
</head>
<body>
  <p>
    Greetings from your Webmaster!
  </p>
</body>
</html>
\end{verbatim}
To Code with Style or Not?

• There’s a fundamental philosophical question at the heart of the significant white space debate: should a language enforce a particular style of writing, or should authors be free to write in whatever style they like?

• Markup doesn’t require significant white space. If you want to add a new line and an indentation every time you nest an element, you can do so, but browsers and validators don’t require it. This doesn’t mean that markup is a free-for-all. Some flavors of markup enforce a stricter writing style than others.

• Before XHTML 1.0, it didn’t matter if you wrote tags in upper-case or lowercase. It didn’t matter whether or not you quoted attributes. For some elements, it didn’t even matter whether you included the closing tag.
To Code with Style or Not? (Cont)

• XHTML 1.0 enforces the syntax of XML.
  
  – **Case sensitivity** All tags must be written in lowercase.
  – **Quoted attribute values** All attribute values must be quoted.
  – **Closing tags** All elements must have a closing tag. For standalone elements such as `br`, the requirement for a closing tag is replaced with a requirement for a closing slash: `<br />`
  – **Explicit attribute values**
  – **id** and **name** attributes
  – **Element nesting** must be proper

• With HTML5, anything goes. Uppercase, lowercase, quoted, unquoted, self-closing or not; it’s entirely up to you.
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

• **Portable Network Graphics (PNG)**
  – Relatively new
  – Should eventually replace both gif and jpeg
Images

• Images are inserted into a document with the `<img />` tag with the `src` attribute.

  – The `alt` attribute is *required* by XHTML (but can be empty string “”)

    » Purposes:

    1. Non-graphical browsers
    2. Browsers with images turned off

    `<img src = "comets.jpg" alt = "Picture of comets" />`

• The `<img>` tag has 30 different attributes, including `width` and `height` (in pixels or %)
Images (continued)

<!-- 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" </h3>
<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 <br />
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
  
  – Note: Relative addressing of targets is often easier to maintain and more portable than absolute addressing
Hypertext Links (continued)

<!-- 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 <br />
<a href = "C210data.html">
   Information on the Cessna 210
</a>
</p>
</body>
</html>
Hypertext Links (continued)

Aidan's Airplanes

The best in used airplanes

"We've got them by the hangarful"

Special of the month

1960 Cessna 210
Information on the Cessna 210

1960 Cessna 210 Information
577 hours since major engine overhaul
622 hours since prop over haul

Buy this fine airplane today at a remarkably low price
Call 999-555-1111 today!
Hypertext Links (continued)

• If the target is not at the beginning of the document, the target spot must be marked

• 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 could 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 it without taking the link should not be made less pleasant

• Links can have images:

```
<a href = "c210data.html"
   
   <img src = "smallplane.jpg" alt = "picture of an airplane " />

   Info on C210

</a>
```
Lists

- *Unordered lists*
  - The list is the content of the `<ul>` tag
  - List elements are the content of the `<li>` tag

```html
<h3>Some Common Single-Engine Aircraft</h3>
<ul>
  <li>Cessna Skyhawk</li>
  <li>Beechcraft Bonanza</li>
  <li>Piper Cherokee</li>
</ul>
```
Lists (continued)

- **Ordered lists**
  - The list is the content of the `<ol>` tag
  - Each item in the display is preceded by a sequence value

```html
<h3>Cessna 210 Engine Starting Instructions</h3>
<ol>
  <li>Set mixture to rich</li>
  <li>Set propeller to high RPM</li>
  <li>Set ignition switch to "BOTH"</li>
  <li>Set auxiliary fuel pump switch to "LOW PRIME"</li>
  <li>When fuel pressure reaches 2 to 2.5 PSI, push starter button</li>
</ol>
```
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

```html
<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
- The `border` attribute can be set to a number, which will be the border width
- Without the `border` attribute, the table will have no lines
- Tables are given titles with the `<caption>` tag, which can immediately follow `<table>`
### Tables (continued)

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

```html
<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 table can have two levels of column labels
  – If so, the `<th>` tag to specify that the label must span some number of columns

```html
<tr>
  <th colspan = "3"> Fruit Juice Drinks </th>
</tr>
<tr>
  <th> Orange </th>
</tr>
```

![Fruit Juice Drinks table]

```plaintext
Fruit Juice Drinks
Orange Apple Screwdriver
```
Tables (continued)

- The **align** attribute controls the horizontal placement of the contents in a table cell
  - Values are **left**, **right**, and **center** (default)
  - **align** is an attribute of `<tr>`, `<th>`, and `<td>` elements

- The **valign** attribute controls the vertical placement of the contents of a table cell

![Table with align and valign attributes](image)
Tables (continued)

- 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 the content of a cell and the inner walls of the cell.

![Table examples](image-url)
### Tables (continued)

```html
<table cellspacing = "50">
  <tr>
    <td> Colorado is a state of ... </td>
    <td> South Dakota is somewhat... </td>
  </tr>
</table>
```
Semester Two Undergraduate Units

**Level One**
- 230.124 Java Programming (Unit Outline)
- 670.104 Software Engineering (Unit Outline)
- 600.105 Computing for Engineers and Scientists (Unit Outline)

**Level Two**
- 670.200 Software Engineering: Design (Unit Outline)
- 230.205 Operating Systems (Unit Outline)
- 230.223 Data Structures & Algorithms (Unit Outline)
- 230.227 Discrete Structures (Unit Outline)

**Level Three/Four**
- 230.302 Artificial Intelligence & Logic Programming (Unit Outline)
- 230.307 Professional Computing (Unit Outline)
- 231.313 Databases (Unit Outline)
- 231.315 Robotics (Unit Outline)
- 231.317 Computer & Network Security (Unit Outline)
Forms

• A form is the usual way to get information from a browser to a server

• XHTML has tags to create a collection of objects that implement 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, or components of a form 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
  – `action =
    – "http://www.cs.ucp.edu/cgi-bin/survey.pl"
  » If the form has no action, the value of `action` is the empty string
Forms (continued)

• The method attribute of `<form>` specifies one of the two possible techniques of transferring the form data to the server, `get` and `post`.
  – you might recognise these in the topic on protocols.

• Widgets

  – Many are created with the `<input>` tag
    » The `type` attribute of `<input>` specifies the kind of widget being created
    • Text
    • Text Areas
    • Checkboxes
    • Radio buttons
    • Menus
    • Reset and submit buttons
Forms (continued)

- **Text**
  - Creates a horizontal box for text input

- **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 checkbox that is not ‘checked’ 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"
Forms (continued)

• Grocery Checklist

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

![Grocery Checklist](image.png)
Forms (continued)

- 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
  - If no button in a radio button group is ‘pressed’, the browser often ‘presses’ the first one

```html
<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>
```
Forms (continued)

- Menus - created with `<select>` tags
  - There are two kinds of menus, those that behave like checkboxes and those that behave like radio buttons (the default)
    - Menus that behave like checkboxes are specified by including the `multiple` attribute, which must be 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` is set to > 1 or if `multiple` is specified, the menu is displayed as a pop-up menu
  - Each item of a menu is specified with an `<option>` tag, whose pure text content (no tags) is the value of the item
    - An `<option>` tag can include the `selected` attribute, which when assigned “selected” specifies that the item is preselected
Forms (continued)

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)

- Text areas - created with `<textarea>`
  - Usually include the `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

```
<form action = "">
  <p>
    <textarea name = "aspirations"  rows = "3"
               cols = "40">
      (Be brief and concise)
    </textarea>
  </p>
</form>
```
Forms (continued)

- 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. Encode the data of the form
  2. Request that the server execute the server-resident program specified as the value of the `action` attribute of `<form>`
Validation

**W3C® Markup Validation Service**
Check the markup (HTML, XHTML, ...) of Web documents

Validate by URI
Validate a document online:
Address: [input field]

› More Options

Check

This validator checks the **markup validity** of Web documents in HTML, XHTML, SMIL, MathML, etc. If you wish to validate specific content such as **RSS/Atom feeds** or **CSS stylesheets** or to **find broken links**, there are **other validators and tools** available.
HTML 5 Validation

http://html5.validator.nu/

Validator.nu (X)HTML5 Validator (Highly Experimental)

Validator Input

Address

- Show Image Report
- Show Source

Validate

About this Service • More options
More References

• Note these may include non-XHTML features!

• Tutorials (check for XHTML compatibility)
  – Getting Started with HTML, by Dave Ragget
    http://www.w3.org/MarkUp/Guide/
  – More Advanced Features, by Dave Ragget
    http://www.w3.org/MarkUp/Guide/Advanced.html

• Reference
  – HTML & XHTML: The Complete Reference
    http://www.htmlref.com/

• Reading
  – Adding a Touch of Style, by Dave Ragget
    http://www.w3.org/MarkUp/Guide/Style.html
• Web 2.0 in 5 minutes ...

http://www.youtube.com/watch?v=6gmP4nk0EOE