XHTML Web Page Authoring

XHTML is the newest version of HTML using HTML to make the specification much more precise, regular and therefore correct

Why do this?

1. Each browser takes a different view of sloppy HTML
2. Each browser adds its own functions to HTML
3. It is much easier to tailor the presentation to different contexts if the basic page structure is well defined
4. Different user agents – e.g. screen readers can make more use of the structure and change the presentation

Some Differences between HTML and XHTML

HTML	XHTML
Starts <html>	Starts with an XML prologue then <html>
Tags are **case insensitive** <EM> = <Em> = <em>	All tags are **lower case** only <em> allowed
Some tags (e.g. <p>) need not be closed	All tags must be **closed** – i.e. must have an end tag
Empty elements not closed <hr>	Empty tags must be closed <hr/>
Hierarchy not enforced	Hierarchy enforced strictly
Attributes ok without quotes size=20	Attributes must be in quotes size="20"
Attributes can be minimised <table border>	No attribute minimisation <table border="true">
Minimal restrictions on placement	**Restricted placement** of elements e.g. no <p> inside <h1>
Style tags (e.g. <font>) ok	Style tags deprecated

Three Flavours of XHTML

XHTML 1.0 Strict

– Only provides structural markup associated with layout
– Cascading Style Sheets (CSS) to get font, colour, and layout effects

XHTML 1.0 Transitional

– Retains some of the style tags and attributes, e.g. bgcolor, text and link attributes

XHTML 1.0 Frameset

– Allows the use of frames to partition the browser window into two or more regions

An XHTML file can be validated by using the W3C validator at:

http://validator.w3.org/

pass it the file name and it checks the flavour specified in the DOCTYPE

HTML and XHTML Elements

In HTML, the text is divided into tagged components called **elements**

Each element is introduced by <tag> and completed with </tag>, where tag indicates the kind of element it is, e.g.:

```
<p>This is a paragraph.</p>
```

– NB - in HTML you can get away without the end tag for some element types (e.g. </p>), XHTML forces this to be present
– Failure to add an end tag is the major reason why some sites work in some browsers and not others

Elements can either be **empty** or have **content**:

– e.g. <hr/> introduces a horizontal line, but has no content, whereas <p>...</p> surrounds some text is the content of the paragraph

Elements with no content need no end tag, but are indicated as empty by having a slash before the right angle bracket (e.g. <hr/>)

Tags in HTML are not case sensitive, but are in XHTML where they **must** be **lower case**

Attributes

Each element may also have attributes added in the open tag, e.g.:

```
<img  src = "X.gif“ alt=“My Picture”/>
```

– introduces an image and has the source of its file as one attribute and some alternative text as another

There are four broad classes of attributes:

Use this

1. **Core Attributes** – available for all entities – identifying attributes (*name* and *id*), style attributes (*class* and *style*) and informational attributes (*title* and *lang*)

not these

2. **Style Attributes** – used to specify specific style information for a particular element – e.g. *background*, *bgcolor*, *text*, *link*, *vlink* and *alink* for the body element – deprecated in favour of style sheets

3. **Event Attributes** – used to call scripts, e.g. when submitting a form

4. **Tag specific attributes** – e.g. the *src* attribute for images

Identifying and Informational Attributes

The *id* attribute **names** an element (it was previously called the *name* attribute) has a special role to play, for instance:

– the name of an **anchor** indicates a point to which the browser can jump

e.g. in file X.html, <a id=”john”> means you can then put

```
<a  href=”X.html#john”>
```

in another document to jump to that point – see Slides 169-170

– the name of a map refers to an **image map** overlaid on an image – see Slide 184

– the name of a form control can be used in the server-side program which uses the form

- e.g. if we add a textbox `<input  id=”address”  type=”text”  size = "80“/>` to a form, whatever is entered in the textbox will be available to the program under the name *address*

The *title* attribute can be used to indicate something the browser can display – e.g. as a tool tip

The *lang* attribute indicates the human language

The Basic XHTML File Structure

Uses strict XHTML

An XHTML file basically looks like:

XML Prologue

```
<?xml version="1.0" encoding="UTF-8"?>
<!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" lang="en">

<head>
<title>A Title for the Browser to refer to</title>
        other information describing the page
</head>
<body>
        The displayable text
</body>
</html>
```

Is in English

Notes on XHTML File Structure

The first lines is the necessary **XML prologue** – all XML files start with this

The second line points to DTD which defines strict XHTML

The third line introduces the document (by using the opening tag **<html>**) and sets the attribute *xmlns* to point to XHTML namespace

Lines 4-7 contain the **file header** (which holds descriptive information about the file) and is delimited by **<head> ... </head>**

Within that (line 5) is a **title** for the page (not displayed but used as an identifier, e.g. in the history) and is delimited by **<title> ... </title>**

The **displayable part of the page** is on lines 8-10 and is delimited by **<body> ... </body>**

The file is terminated on line 11 by the ending **</html>** tag

Comments and Special Characters

Comment lines look like this: `<!-- this is a comment -->`

- Comments start with "<!--" and only end when you reach "--".
- So they can span several lines and they can end unexpectedly for some browsers, e.g. in :

  `<!-- important -- this is a comment -->`
- the comment ends after "important".

If you need to include characters that HTML uses for syntax, you must escape these characters (c.f . "\" in Java) using what XML calls an **entity**

This is done by describing the character between "&" and ";", e.g.

&lt; and &gt; - "<" and ">"	&reg; - the trademark symbol
&quot; - double quote	&copy; - the copyright symbol
&nbsp; - a non-breaking space	&amp; - "&"

NB – the ASCII value can also be used - &#061; also creates "A".

The <head> Section

So far, we have just seen one component of the <head> section (the title).

Here are a few others:

<meta> This adds meta-information to the page such as the author, but it can also be used to achieve some sophisticated effects, such as :

- adding keywords for search engines

  `<meta http.equiv="keywords" content = "Seattle Popgroup Green Pajamas">`
- getting the page automatically refreshed every so often - "client pull"

  `<meta http.equiv="refresh" content = "60">`

<style> This defines a style, similar to a word processing style which can be used to determine the look of your page - see later in course

<base> This determines the place where the browser looks first to place targets – particularly for frames

Block Elements

The body of the document is broken up into a series of elements called *block* elements, which have the following features:

- each start on a new line when rendered
- embedding one inside each other is sometimes wrong (e.g. <p> in <h1>)
- all text must be inside a block element
- inline elements (e.g. hyperlinks) go inside block elements

These are the block elements:

<p> a paragraph with a blank line before and no indentation
<div> a larger division of the text, c.f. a section in Word
<h1> .. <h6> increasingly small headings
<blockquote> indented quotations
<pre> preformatted text that you don't want the browser to change, e.g. poems
<br/> an empty element forcing a new line
<hr/> an empty element placing a horizontal rule next

Inline Elements

These allow separate formatting of fragments in a block. There are four kinds:

- **Physical style tags** (**absolutely not to be used as they go against the whole purpose of XHTML**)

<b> - bold	<i> - italics	<big> - font	<small> - font
<u> - underline	<tt> - fixed width font		<strike> - strike through
<sub> - subscript	<sup> - superscript		<blink> - never use this!!!

- The **<font>** tag allows the text to be rendered in a different font or size – also deprecated
- **Logical style tags** – these are ok

  <em> - emphasise (usually italics)
  <strong> - emphasise even more (usually bold)
  <code> - for computer code (usually in fixed width font)
  <cite> - for a citation
  <abbr> and <acronym> - use the title tag to indicate full meaning
- The <span> tag, which identifies a fragment of text for formatting or identification purpose, c.f. <div>

Lists

HTML supports the display of two kinds of indented list:

<ul> - means an **unordered list**, whose elements are introduced with <li>

<ol> - indicates an **ordered list**, elements are also introduced with <li>
NB – bullet and number styles can be varied with an attribute

<li> - indicates an **item** within a list

<dl> - introduces a **definition list**, whose elements are pairs of terms (tagged with **<dt>)** and definitions (tagged with **<dd>**)

Lists can be nested, example:

```
<ul>
      <li>list A
      <ol>  <li>two list</li>
             <li>three list</li>
      </ol>
      </li>
      <li>list B </li>
</ul>
```

produces

- list A
    1. two list
    2. three list
- list B

The style attribute – *list-style-type* can be used to change the bullet character or the numbering style

Embedded Objects

Images - the tag <img> introduces a link to a file which is an image and which will usually be displayed as part of the file

– they **must** have an attribute defining the source of the file - *src*

– they **must** also have an attribute specifying text to be displayed if images are not available - *alt*

– they **may** have attributes defining the *width* and *height* of the area to be displayed in

e.g. `<img   src = "MyPic.gif"   alt = "My Picture"  width="300"/>`

displays an image in a file called *MyPic.gif* scaled so that the width is 300 pixels

More generally, the <**object**> tag is used to introduce non-text data

`<object data="penguins.mov" type="video/quicktime" width="600"/>`

- Note the use of the MIME type to specify the type
- Note also that this feature is not supported on all browsers

Tables

Tables embed a rectangular table into the text

– these are primarily used to allow tabular data to be displayed

– they are also used to align objects (after suppressing the border)

– indeed table are often used to create sophisticated pages with various kinds of content placed in the cells

The element types involved are:

– the <table> tag is used to introduce the table

– the <tr> tag indicates a row of the table

– the <th> tag indicates a cell in the table which is a heading (and probably emboldened)

– the <td> tag indicates an ordinary cell in the table:

Not really needed as browser can work these out

```
<table rows = "2" cols = "3" border = "true">
    <tr> <th>Col 1</th>   <th>Col 2</th>   <th>Col 3</th>   </tr>
    <tr> <td>Cell 11</td> <td>Cell 12</td> <td>Cell 13</td> </tr>
    <tr> <td>Cell 31</td> <td>Cell 32</td> <td>Cell 33</td> </tr>
</table>
```

Hyperlinks

A hyperlink in an HTML document is indicated by an **anchor**

This is tagged with <a> and includes a attribute, "href", indicating where the linked document is to be found

Thus the following:

`<a  href = "LinkedFile.html">  Link to File  </a>`

– displays "Link to File" to show that it is a link. Selecting the link goes to the file called *LinkedFile.html* in the current directory.

An example:

```
<table cols = "2" border = "true">
<tr> <th>Student</th>  <th>Project Description</th></tr>
<tr>   <td> <a href = "mailto: ablej@dcs.gla.ac.uk">John Able<a> </td>
        <td> <a href = "john.html">John's Project</a> </td>   </tr>
......
</table>
```

Targets or Uniform Resource Indicators

The value given to the **href** attribute is called the **target**

This is more general than just a URL and is called a **Uniform Resource Indicato**r (URI). It can be any of:

- a URL – e.g. “http://www.dcs.gla.ac.uk/”
- a file in the server's file system – e.g. "LinkedFile.html“ which means that the file *LinkedFile.html* is in the same directory as the file holding the current page
- **Note – under no circumstances make a file point to a fixed location in the file directory – all references should be relative**
- a named point in a document (in the current document this allows a jump within a document)

The latter is achieved by:

i) making a hyperlink reference. e.g.: <a href="#jumpHere">

ii) defining a point to jump to - the target, e.g.:<a name = "jumpHere" />

Example - an Alphabetic Index

An alphabetic index is commonly used at the top of a page - selecting each letter jumps to the first item in the document starting with that letter:

A B C ..... Z

i) At the top of the file put:

```
<a href="#A">A</a>
<a href="#B">B</a>
.....
<a href="#Z">Z</a>
```

ii) Throughout the file, put lines of the form, e.g.:

```
<a name = "A“/>
<p>Alan</p>
<p>Andrea</p>
<a name = "B“/>
<p>Bill</p>
```

Forms

A form in HTML permits the input of data and its transmission to a program on the server or by e-mail to another user

The basic structure of an HTML form is:

```
<form action = "url" method = "POST">
        a set of HTML "controls" to describe the content
</form>
```

The effect of this is to send the data in the form to a program located at the URL when the form is submitted by use of a submit button

The URL will either refer to a program or be a *mailto* URL

The method is either "GET" or "POST" - see later in the course for the difference, but roughly GET is for retrieving data and POST is for storing data

XHTML Controls

Controls are the individual components of the form which allow user interaction

There are several kinds:

input controls specify controls which allow data entry

button controls place a button on the form which can be pressed to cause an action to take place

select, **optgroup** and **option** controls place a pop-up menu on the form

textarea controls place a multi-line text entry area on the form

labels - permit (non-interactive) labels to be added to the form

Input Controls I

An input control allows the user to enter values which are then sent when the form is submitted.

For example:

```
<p>Surname: <input  type = "text"  id = "Sname"  size="30“/></p>
```

This puts up a one line text entry box, labelled "Surname:" which shows up to thirty characters and returns the data entered paired with the variable name "Sname“. Note it has no character content in the XHTML file

In the above, it is important to distinguish the following:

Surname: This is part of the displayed text on the page and has nothing to do with the processing of the data

Sname: Is the name of the text box and will be used by the program processing the form as a parameter name

- i.e. “*Sname = data typed in”*   will be sent to the server

Input Controls II

The main attribute of an input control is its *type* - the default being text, as above

These are the main ones:

password - a text box that echoes characters as stars to hide the input

checkbox - an on/off switch, useful for yes/no decisions

radio - an on/off switch, but one which is grouped with other switches by use of the same name, so that only one is on at any time, e.g.:

```
<input  type = "radio"  name = "sex"  value = "male“/>
<input  type = "radio"  name = "sex"  value = "female“/>
```

hidden - not a displayed form element but a means for the form to submit data not entered by the form user, e.g. session identifier

submit - puts up a button, which causes the form to be submitted

reset - puts up a button, which causes the form to be reset to its original state - for both buttons, the value attribute determines the label on the button

Other Controls I

Button controls duplicate the submit and reset input controls or provide a means of invoking a script

Textarea controls provide multi-line input, e.g.:

```
<textarea   name="Description"   rows="20"   cols="78“/>
```

– This provides for the textual input of a variable named "Description" using 20 character rows and 78 columns

Select controls are used to give pop-up menus of the optional values of a form element and **option** controls are used to list the options, e.g.for a title field:

```
Title: <select name="Title">
      <option>Dr.</option>  <option>Prof.</option>
      <option>Mr.</option>  <option>Mrs.</option>
      <option>Ms.</option>  <option>Miss</option>
</select>
```

Other Controls II

You can provide a default, permit multiple selections or determine the number of option visible without scrolling

Optgroup controls are used to group the options into a hierarchy, e.g.:

```
Title: <select name="Title">
      <optgroup label = "Male Titles" size = "3">   // all 3 visible
                  <option>Dr.</option>  <option>Prof.</option>
                  <option selected = "true">Mr.</option>     // default
      </optgroup>
      <optgroup label = "Female Titles" size="5"> // all 5 visible
                  <option>Dr.</option>  <option>Prof.</option>
                  <option selected = "true">Ms.</option>     // default
                  <option>Mrs.</option>  <option>Miss</option>
      </optgroup>
</select>
```

Labels are used to decorate the form with textual information.

Form Example I

A form to pick up project proposers, titles, names, emails and departments; and project titles and descriptions:

```
<html><head><title>Project Proposal Form</title></head>
<body style = “background-color : #F0FFFF">
<h1 style = “text-align : center; color : #006600">
    Submitting a Project Proposal<br/>for the Msc IT</h1>
<hr/>
<p align=center>Please enter the information and press submit.</p>
<hr/>
<form method="POST" action="formscript.pl">
<table>
<tr> <td>Title:</td>
        <td> <select name="Title">
            <option>Dr.</option>  <option>Prof.</option> <option>Mr.</option>
            <option>Mrs.</option> <option>Ms.</option>  <option>Miss</option>
    </select> </td> </tr>
<tr> <td> Forenames: </td><td> <input name="Forenames"   type="text"
    size="24"></td> </tr>
<tr> <td> Surname: </td><td> <input name="Surname"   type="text"
    size="24">   </td> </tr>
```

Form Example II

```
<tr> <td> Department: </td><td> <select name="Dept">
        <option>Computing Science</option>
        <option>Electronics and Electrical Engineering</option> ......
</select></td> </tr>
<tr> <td> Full e-mail address: </td><td> <input name="Email1" type="text"
    size=“30"> </td> </tr>
<tr> <td> <strong>Project title</strong> </td><td> <input name="ProjTitle"
    type="text" size="79"></td> </tr>
<tr> <td> <strong>Description</strong> </td> <td>
    <td> <textarea name="Description" rows="20"   cols="78" style =
    “background-color : #FF00FF"></textarea> </td> </tr>
</table>
<input  value="Submit proposal"  type="submit“/>
<input  value="Clear form completely"  type="reset“/>
</form>
</body>
</html>
```

Page Layout

There are three techniques used for dividing a page into panels. In order of age and increasing quality, these are:

frames

a set of tags specifically for this purpose

tables

using the basic table structure to position and align objects

div and **span**

defining the structure of the page and identifying position using the separate style facilities

```
<div id="leftColumn">
        X1
        X2
</div>
```

in the style sheet

```
#leftColumn {float: left; width: 20%;}
```

Frames

A frame is a subdivision of the page whose contents are independent of other frames on the page.

Frames are tagged using <frameset> which is used instead of <body>, e.g.:

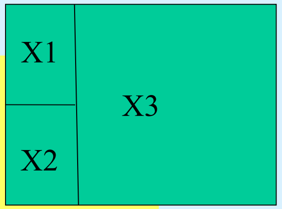
```
< frameset cols = "20%, 80%"   rows =  "20%, 30%,50%">
```

which divides the page into six frames (2 columns and 3 rows)
the first column occupies 20% of the width, the second 80%, etc.

There should then be six entries of the form: <frame src="X.html"> before the </frameset> ends the frame specification

Example:

```
<frameset  cols = "20%, 80%">
    <frameset  rows = "50%, 50%">
      <frame src = "X1.html“/>
      <frame src = "X2.html“/>
    </frameset>
    <frame src = “X3.html“ id=“Content”/>
    <noframes> This is best viewed using frames</noframes>
</frameset>
```

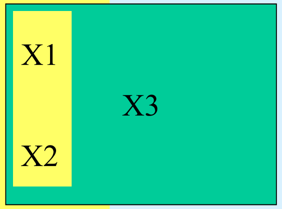


Tables not Frames!

The use of frames is controversial as the forward and back buttons can have strange effects

Therefore, many web sites now use tables for a similar effect:

```
<body style="background-color : #ff00ff">
<table border = "0" style="background-color : #00ff00">
<tr>
<td>
<table border = "0" style="background-color : #ffff00">
<tr><td>X1</td></tr>
<tr><td>X2</td></tr>
</table>
</td>
<td>X3</td>
</tr>
</table>
</body>
```

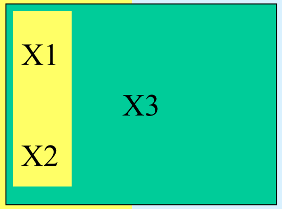


Divs not Tables!

But tables are not the best option
- They are slow to render
- The layout of a page is not conceptually a table

So use divs and style sheets

```
<body style="background-color : #00aa00">
<div id="content">
<div id="leftcolumn" style="float:left; background-color : #ffff00">
<p>X1</p>
<p>X2</p>
</div>   <!-- End of leftcolumn -->
<div id="rightcolumn"  style="float:right" >
   <p>X3</p>
</div>   <!-- End of rightcolumn -->
</div>   <!-- End of content -->
</body>
```



The Separation of Style

In the previous slide, the style attribute was used to decide where elements went and how they look

The proper way to do this is to use a style sheet and the id attribute to link the two

The previous example should not have style attributes but link to a style sheet containing:

```
body { background-color : #00aa00 }
#leftcolumn  { float:left;  background-color : #ffff00 }
#rightcolumn { float:right }
```

This is a **cascading style sheet** to be discussed in the next lecture

HTML Image Maps

An image map is an image file over which has been superimposed a number of touch sensitive areas.

The areas are described as polygons or circles and clicking over these results in moving to another location. Example – touch sensitive map of Europe at:

http://www.eurochild.gla.ac.uk/Documents/UN/StatePartyReports/Map.htm

Italy
Vug
Moldova
Bulgaria
Alb
Mac
Portugal
Spain
Vatican
Slovenia
Greece
Turkey
Gibraltar
Malta
Bosnia-H
Croatia
Cyprus
©1996 Brandon Plewe

```
<img src="Euromap.gif" usemap="#EuroMap">        <!-- means use the gif file
   as backdrop and the map information as the selectable foreground -->
<map name="EuroMap">
   <area shape="circle" coords ="234, 282, 3"  href="Index.htm#Holy See">
   <area shape="poly" coords ="11, 296, 49, 295, 49, 293, 54, 273, 77, 281,
         63, 324, 47, 325,47, 311, 10, 307" href="Index.htm#Portugal">
</map>
```