CSCC09F
Programming on the Web

XHTML
eXtensible HyperText Markup Language

an XML application, global structure, head element, content elements, frames, forms, linking to style & scripts

<html>

- XHTML is an XML application (aka vocabulary, tagset)
- XML is case-sensitive.
  - elements and attributes in the xhtml namespace are lower-case
  - differs in this respect from HTML 4.0
- Definitive references:
  - html 4.01 specification 24 Dec. 1999
    - http://www.w3.org/TR/html4/
  - xml 1.0 (4th edition) specification, 16 Aug, 2006
    - http://www.w3.org/TR/xml/
  - xhtml 1.0 (2nd edition) specification, 1 Aug 2002
    - http://www.w3.org/TR/xhtml1/
  - html 5 (editor's draft), 10 Sept, 2014
    - http://dev.w3.org/html5/spec/Overview.html

Sections

- Global Structure (head): title, meta, link, base, etc.
- Text: alignment, phrase elements, sections etc.
- Links: anchors and fragment ident's
- Lists: unordered, ordered, labelled
- Tables: alignment and data
- Forms
  - User-input Forms: Text Fields, Buttons, Menus, etc.
- Frames: multi-view presentation of documents
- Style Sheets: linking, covered later as CSS
- Scripts: linking, covered later as JS and DOM

example: global.xml

Global Structure <head>

- Must contain a **<title>** element
  - used as window titles, in favourite lists, search engine results, …
  - choose a good, context-rich title – it gets displayed a lot
    - POOR: **<title>Intro</title>**
    - BETTER: **<title>Intro to Medieval Bee-Keeping</title>**
- May contain
  - **<meta>**
  - **<link>**
  - **<base>**

Global Structure <meta>

- Uses include:
  - general info, e.g. author, description (used by search engines)

```
<meta name="author" content="Alan Rosselet" />
<meta name="description" content="XHTML overview" />
```

  - document-level HTTP content control, e.g. scripting language, caching, refreshing (and client “pull”), expiration.

```
<meta http-equiv="Content-Type" content="text/html;
      charset=utf-8“ />
<meta http-equiv="Content-Script-Type"
      content="text/javascript“ />
<meta http-equiv=“cache-control"
      content="no-cache" />
<meta http-equiv="refresh"
      content="10; url=http://www.google.ca" />
<meta http-equiv="expires"
      content="Fri, 21 Sep 2012 18:22:38 GMT" />
```

Global Structure <meta>

- Uses include:
  - instructing search engines on how to treat your site (e.g. keywords, revisit-after – for material that changes frequently, distribution-locality, robots & googlebot – for index control)

```
<meta name="keywords" lang="en"
                content="vacation, sunshine“ />
<meta name="keywords" lang="fr"
                content="vacances, soleil“ />
<meta name="revisit-after" content=“30 Days” />
<meta name=“distribution” content=“IU” />
<meta name=“robots” content=“index, nofollow” />
```

- Efforts underway to standardize document meta-data
  - e.g., Dublin Core Metadata Initiative

Global - <link>

- Associate external stylesheets with a document (see also <style> slide below)
- Conveys relationship information that may be rendered by user agents in a variety of ways or used by search engines
  - e.g. a tool-bar with a drop-down menu of links, to find subsections of a document

```
<head>
  <title>General Relativity - Chapter 2</title>
 <link rel="stylesheet" type="text/css" href="c09.css"/>
 <link rel="shortcut icon" href="favicon.ico" />
  <link rel="Index" href="../index.html“ />
  <link rel="Next" href="ch3.html“ />
  <link rel="Prev" href="ch1.html“ />
  <link rel="Start" href="grtitlepage.html“ />
</head>
```

Global - <base>

- Specifies an absolute URI that serves as a base URI for all relative URI's in the document, eg:

```
<html>
  <head>
    <title>PR Pages for U of T</title>
    <base href="http://www.utoronto.ca/pr/"/>
  </head>
  <body>
   <p>Current edition of
         <a href="bulletin">UT Bulletin</a></p>
  </body>
</html>
```

- Supports page portability – move document tree of Web pages and objects such as images by simply redefining value of base.
- example: global.xml

%attrs; = Common Attributes

- Certain attributes are common across many elements
  - %coreattrs; = core attrs
    - id (e.g. style selector, link anchor, script ref, object name),
    - class (e.g. for style selection, binding jQuery behavior),
    - style (css attributes),
    - title (small pop-up box on mouseover)
  - %events; = scripting events
    - onclick, ondblclick, onmousedown, …
  - %i18n; = internationalization attrs
    - lang language code, e.g. …
    - dir (ltr|rtl) e.g. <bdo dir="rtl">…</bdo>
    - example: i18n.xml
  - Many commonly-used attributes in older HTML versions are deprecated in favour of stylesheets.

example:
text.xml

Text - Headings

- XHTML includes six levels of section headings, with **<h1>** as the most important and **<h6>** as the least.
  - Visual browsers usually render more important headings in larger fonts than less important ones.
- Note, headings do not surround their section, only their titles.
  - ** ** or other block-markup elements typically used to bracket the text content of the section

Grouping Elements: Section Delimiters

- Used in conjunction with class and id attributes to associate stylesheet information and JavaScript behavior with document elements
  - **`<div>`**
    - generic “block-level” element (implicit line break)
  - **`<span>`**
    - generic “inline” element (no line break)
- No visible rendering effect is associated with either, other than line break with **`<div>`**, when used without style information.

Section Delimiters

```
<div class="section" id="forest-elephants">
  <h1>Forest elephants</h1>
  <p>In this section, we discuss the lesser
  known <span class="key-topic">forest
  elephants</span>.</p>
  ...this section continues...
  <div class="subsection" id="forest-habitat">
    <h2>Habitat</h2>
    <p>Forest elephants do not live in trees
    but among them.</p>
    ...this subsection continues...
  </div>
</div>
```

example: lines.xml

Text - Lines and Paragraphs

- **<p>** starts a new, logical paragraph.
  - may contain only inline elements
  - no block elements allowed (so e.g. no nested paragraphs)
- **
** forces a line break
  - clear attribute indicates where the next line should start
- An non-breaking space prohibits a line break
  - ** ** ** **
- soft hyphen encourages a line break
  - **** ****
- example: lines.xml

Text - Phrase Elements

example:
phraseelements.xml

- Phrase elements add structural information to text fragments.
- example: phraselements.xml

example: quotes.xml

Text - Quotations

- **<blockquote cite="http://www.w3.org/…" >**
  - for a block of text
  - indents the entire paragraph
  - ought to add " signs, but doesn't because of historical use as simply an indented paragraph
- **<q>**
  - for in-line quotes
  - can be nested
  - doesn't work fully in current browsers (does not use matching quote char's and does not switch quote char's for nested instances)
- example: quotes.xml

examples:
supb.xml
insdel.xml
pre.xml

Text - misc.

- subscripts and superscripts
  - supb.xml
- inserts and deletions
  - insdel.xml
- preformated text
  - pre.xml

example: links.xml

Hypertext Links

- Created with <a> "anchor" tag
  - Can tag a certain part of the document with an anchor id:
    - `<p id="chapter1"/>`
    - `<div id="chapter1"> … </div>`
  - Can then "jump" there using a fragment identifier:
    - `<a href="#chapter1">go to chapter 1</a>`
    - `<a href="http://www.foo.com/book.html#chapter1>`
  - Used by Backbone.js to create URL's for different app views
- Also use anchors to link entire documents:

```
<a href="http://www…/mydoc.html">mydoc.html</a>
```

example: lists.xml

Lists

- Unordered lists

```
<ul><li>item 1</li><li>item 2</li>… </ul>
```

- Ordered lists

```
<ol><li>item 1</li><li>item 2</li>… </ul>
```

- Definition lists

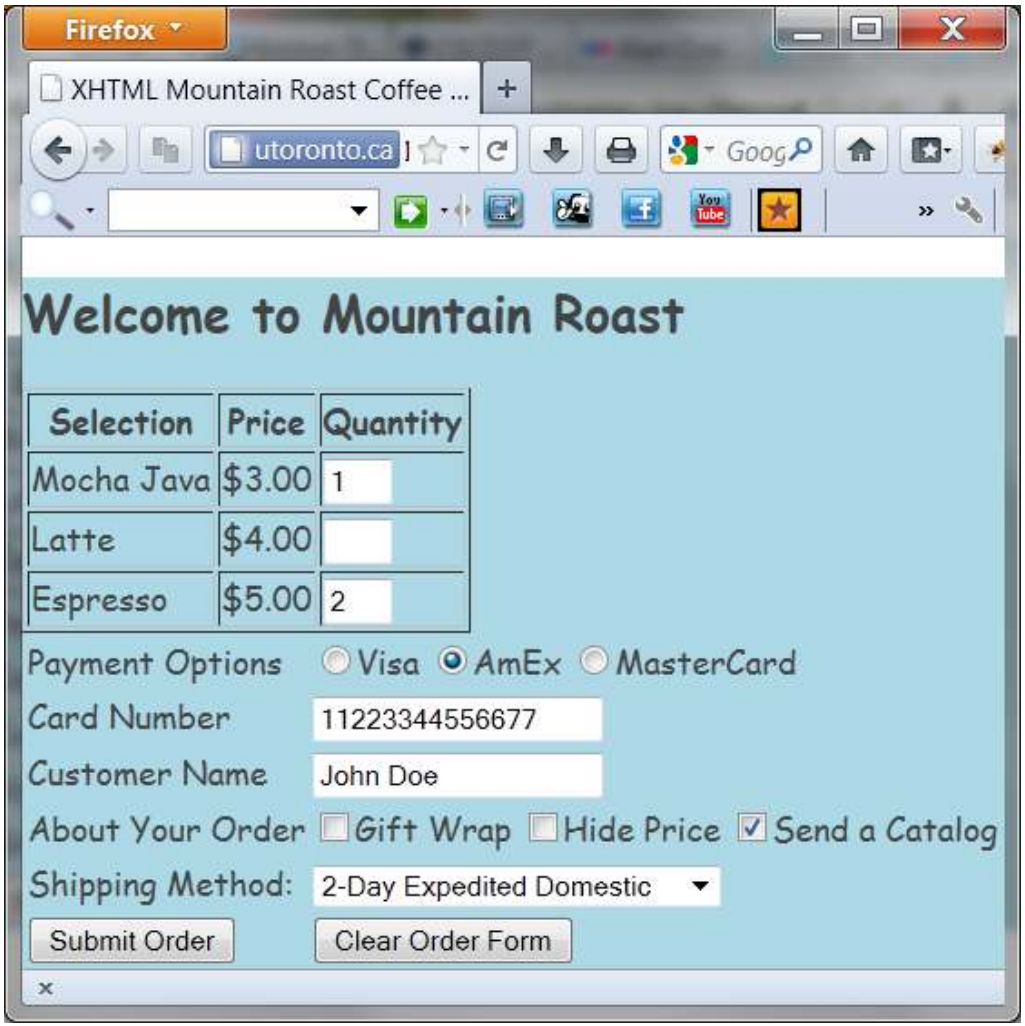
```
<dl>
    <dt>name</dt>
    <dd>text about name</dd> …
</dl>
```

examples: earth.xml, basictable.xml

Tables

- Extremely useful for controlling layout (but CSS is the proper way to control layout)
  - earth.xml
  - You can even use them for laying out tables of data!
    - basictable.xml
    - note the use of spans (rowspan, colspan) to spread data across multiple rows/columns
- Features in latest version of XHTML, e.g.:
  - row and column groups
    - to apply the same style across groups of columns
    - for rows: header, body and footer sections
  - rules (lines between data cells)
  - controlling width

Form Example



example:
form-demo.xml

example:
form-demo.xml

Forms

- Collect user input
  - for processing by client-side scripts
  - for transmission to server-side programs
- Built using widgets/controls, such as:
  - text boxes
  - checkboxes
  - pull-down select menus
  - radio buttons
  - submit and reset buttons
- Each widget/control has a value
- On submit, all widget val's collected and sent to server

Forms - <form>

- **<form>** element encompasses entire input-form structure

- **<form>** attributes:
  - action
    - identifies URL of server-side program to be invoked in response to submit button press
    - e.g. **action="pay.cgi"**
  - method
    - either “GET” (default) or “POST” – more on these later, when we study the HTTP protocol and RESTful servers

Forms - <input>

- <input> element encompasses these widgets/controls:
  - text boxes, checkboxes, radio buttons, user-defined buttons, reset and submit buttons
- attributes:
  - type:
    - text, password, checkbox, radio, reset, submit, hidden, image, button, … (see w3c docs)
  - name, id – for referencing e.g. by script
  - value
    - default for checkboxes and radio buttons
- e.g.:

```
<input type="radio" id="cardtype"
   name="cardtype" value="amex" />
```

Forms – <select>

- **<select>** element encompasses:
  - menus
- attributes:
  - name, id – for referencing, e.g. by script
  - size – for drop down and scrolling lists
  - multiple – multiple values selectable
- **<option>** element children of **<select>** define the allowed values for the menu
- e.g.:

```
<select>
    <option value="1">Credit Card</option>
    <option value="2">Pay Pal</option>
</select>
```

Forms – submit and reset types

- **`input type="reset"`**
  - clears all form fields
- **`input type="submit"`**
  - encode all widget/control values into a string (essentially a parameter list)
  - pass that string (parameter list) to an application on the server, as specified in **`action`** attribute
- more detail and examples of forms when we look at server-side programming

Forms Example

```
<form action="https://mathlab.utsc…" method="POST">
  <h2>Welcome to Mountain Roast</h2>
  <table border="1">
     …. <tr>
      <td>Mocha Java</td><td>$3.00</td><td><input
       type="text" size="3" name="Mocha Java" /></td> …..
  <table>
    … <tr><td>Payment Options</td>
        <td>
          <input type="radio" name="payment" value="visa"/>
              Visa
          <input type="radio" name="payment" value="amex"/>
              AmEx
          …..
          <input type="submit" value="Submit Order"/>
```

Forms example: on submit event

- encode all widget values into a "URL-encoded" string (a string-encoded list of parameter values)
- pass the parameter values to a server-side program for processing
- Observe the URL that appears in your location window when you hit the "Submit Order" button if using GET method e.g.:
  - **…/form-demo.xml?mjava=1&latte=2&espresso=1&payment=amex&number=01230123&name=John+Doe**
- server-side program returns a new Web page to the browser in response, this page replaces the calling page

Frames and iFrames

- HTML frames allow authors to present multiple documents within a single Web page
  - offer designers a way to keep certain information visible, while other views are scrolled or replaced
  - may be independent windows or subwindows
  - frameset element takes the place of body element (though body may still be present for non-frame-capable browsers).
  - many consider use of frames to be bad practice, why?
  - dropped in HTML5
- iFrames are elements that embed one document within another (kept in HTML5)
- examples: frames.xml iframe.xml

Style Sheets

- defined inline using <style> element
- or defined externally and referenced as:
  - `<link rel="stylesheet" type="text/css" href="resume.css"/>`

  or
  - `<?xml-stylesheet type="text/css" href="resume.css"?>`

  or
  - `<?xml-stylesheet type="text/xsl" href="resume.xsl"?>`
- more details later, in CSS lecture slides

Scripts

- declared using **<script>** element
- script may be defined internally within the **<script>** element or **<script>** may provide a reference to an externally defined script as in this example:

```
<script type="text/javascript" src="code.js">
</script>
```

- recall requirement to set default scripting language for intrinsic events:

```
<meta http-equiv="Content-Script-Type"
                content="text/javascript">
```

- more details later, in JavaScript lecture slides