Hypertext markup language HTML

BWA Slides #3

What is the World Wide Web?

"The World Wide Web (Web) is a network of information resources. The Web relies on three mechanisms to make these resources readily available to the widest possible audience:

- A uniform naming scheme for locating resources on the Web (e.g., URIs).
- Protocols, for access to named resources over the Web (e.g., HTTP).
- Hypertext, for easy navigation among resources (e.g., **HTML**)."

//HTML 4.01 Specification

URI vs URL vs URN

URI classifies an

- Uniform Resource Locator, URL, a location
- Uniform Resource Name, URN a name
- ...or both..

URI Examples

- URL: file:///home/hajo/courses/da
- URN: ISBN 0486275574

Some say URI others URL ...confusing...

- HTML5 makes a "will-full" violation of URL (other definition).
- We use URL **or** URI ... so no distinction...

URL is a Complicated Creature

General form (each part can be further dissected)

```
scheme://username:password@domain:port/path?query_string#fragment_id
```

Normally case sensitive

Allowed chars: aA-zZ, 0-9 - _ . ~
Reserved chars: ! * ' ( ) ; : @ & = + $, / ? % # [ ] (have special meaning)

- Reserved chars in between need to be URL encoded (percent encoded)
  - Character used for some other purpose (than default)
- Example Ladies + Gentlemen -> Ladies%20%2B%20Gentlemen

Absolute and Relative URLs'

If scheme part missing should be interpreted as a relative URL

- Browser sometimes (often) imply http://
- Interpreted as relative to the base URL of the document (base URL = URL to document itself)
- If leading / relative to server root
- Example
  - Base URL: http://www.server.example/xyz/bar/zap.html
  - Relative URL: URL "foo.html" in page zap.html
  - Result (as interpreted by server): http://www.server.example/xyz/bar/foo.html

Details in http://tools.ietf.org/html/rfc3986

HTML

"To publish information for global distribution, one needs a universally understood language, a kind of publishing mother tongue that all computers may potentially understand. The publishing language used by the World Wide Web is HTML (from HyperText Markup Language)."

//HTML 4.01 Specification

HTML Confusion

First drafts by Tim Berners-Lee at CERN in late 1991

Then versions up to HTML4.01 (current dominating)

Then XHTML1.0 as a stricter alternative to HTML 4.01 (XHTML is an XML document)

Then XHTML2.0 and HTML5 was announced

- XHTML2.0 cancelled (not backward compatible)

Then XHTML5, which is an update to XHTML1.x, defined alongside HTML5 in the HTML5 drafts ?!?!?

- Also HTML 5.1 (Nightly)

HTML5 vs XHTML5

Put simple: XHTML5 is HTML written in XML, but ...

XHTML tags in lowercase, must use namespace (see XML...)

Some features can't be used

- HTML, no name spaces and more...
- XHTML, "noscript" can't be used and more...

DOCTYPE

- HTML, mandatory
- XHTML, may (must have XML prolog)

and more...

HTML 5

Specification not finished and there are two non-identical (and also 5.1)!!!

- A W3C draft at http://dev.w3.org/html5/spec/Overview.html
- A WHATWG draft at http://www.whatwg.org/specs/web-apps/current-work/ (Apple, Mozilla, Opera)

More HTML5 Confusion

HTML5 is also used as a generic term for;

- Web pages no longer need to look (and act) like web pages
- Web pages no longer need to represent one person/organization's content
- Web pages can function intelligently and easily across display devices

This involves CSS and JavaScript, more to come...

Polyglot HTML

"The language used to create documents that can be parsed by both HTML and XML parsers is called polyglot markup"

- Both valid HTML and well-formed XML
- Identical DOMs when processed as HTML and when processed as XML

http://www.w3.org/TR/2012/WD-html-polyglot-20121025/

How to Survive?

Can't be too picky about this right now

As noted before we prefer XHTML5 or polyglot HTML5

- NetBeans have templates, close to this...

Polyglot HTML5 Page

Example (also using some new semantic tags instead of s', more to come...). In some .html file

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" lang="" xml:lang="">
<head> <!-- Invisible, info/data part (this is a comment) -->
 ...
</head>
<body>                          <!-- Visible content -->
    <header>
        <hgroup></hgroup>
        <nav></nav>             <!-- Menus,or alike -->
    </header>
    <section>

    </section>
    <aside></aside>             <!-- Sidebar -->
    <footer></footer>
</body>
```

HTML 5 (5.1) vs HTML4.01

Much more of highly interactive applications and media in 5

- An API for playing of video and audio which can be used with the new video and audio elements (replacing plugins like: Flash, MS Silverlight, JavaFX,...)
- An API that enables offline Web applications.
- An API that allows a Web application to register itself for certain protocols or media types.
- Editing API in combination with a new global contenteditable attribute.
- Drag & drop API in combination with a draggable attribute.
- API that exposes the history and allows pages to add to it to prevent breaking the back button.
- New elements (and attributes) ; video, audio, canvas and more
- Changed and absent elements (and attributes)

Summary: HTML5 rocks, ... it's cool, ...

Details at http://www.w3.org/TR/html5-diff/

HTML 5.1 Specification Overview

Common infrastructure
The conformance classes, algorithms, definitions, and the common underpinnings of the rest of the specification.

Semantics, structure, and APIs of HTML documents
Documents are built from elements. These elements form a tree using the DOM. This section defines the features of this DOM, as well as introducing the features common to all elements, and the concepts used in defining elements.

The elements of HTML
Each element has a predefined meaning, which is explained in this section .... This includes large signature features of HTML such as video playback and subtitles, form controls and form submission, and a 2D graphics API known as the HTML canvas.

Loading Web pages
HTML documents do not exist in a vacuum — this section defines many of the features that affect environments that deal with multiple pages, such as Web browsers and offline caching of Web applications.

Web application APIs
This section introduces basic features for scripting of applications in HTML.

User interaction
HTML documents can provide a number of mechanisms for users to interact with and modify content, which are described in this section, such as how focus works, and drag-and-drop.

The HTML syntax **The XHTML syntax**

All of these features would be for naught if they couldn't be represented in a serialized form and sent to other people, and so these sections define the syntaxes of HTML and XHTML, along with rules for how to parse content using those syntaxes.

Rendering
This section defines the default rendering rules for Web browsers.

HTML Elements Overview

Categories

4.1 The root element
4.2 Document metadata
4.3 Scripting
4.4 Sections
4.5 Grouping content (common tags)
4.6 Text-level semantics (common tags)
4.7 Edits
4.8 Embedded content
4.9 Tabular data (common tags)
4.10 Forms (common tags)
4.11 Interactive elements
4.12 Links

Global Attributes

Attribute for any element

- Global attributes

Attributes id and class important

- Used in conjunction with style sheets (which elements are affected)
- id's must be unique in page

HTML Tutorials

Will not go into details building basic HTML pages

Many tutorials on Web http://www.w3schools.com/html/

A note on white space: In general, a single whitespace character, including newlines, or a sequence of whitespace characters are treated as a single space and leading/trailing whitespace is eliminated.

HTML Form Elements

"A form is a component of a Web page that has form controls, such as text fields, buttons, checkboxes, range controls, or color pickers. A user can interact with such a form, providing data that can then be sent to the server for further processing (e.g. returning the results of a search or calculation). No client-side scripting is needed in many cases, though an API is available so that scripts can augment the user experience or use forms for purposes other than submitting data to a server.

Writing a form consists of several steps, which can be performed in any order: writing the user interface, implementing the server-side processing, and configuring the user interface to communicate with the server. " // Html 5.1/4.10.1

Form Controls

Input controls have the name attribute (name of value) and more...

Each input control has both an initial value and a current value (strings)

- Initial may be specified with value attribute
- Current replaces the initial (at user input)

If form is reset, each control's current value is reset to its initial value

At form submission (request to server) some name attributes are paired with their current value a included in the submission (aka successful controls)

Basic Form

```
<form action="..." method="..." >
   <!-- Controls -->
   <input type="text" name="   key" value="default"
/>
   <input type="submit" value="Submit" />
</form>
```

action = form processing agent URI (something on server handling the request)
method = HTTP method POST (GET is possible but bad, don't use)
content-type = Media type of request body (post only), default application/x-www-form-urlencoded, for binary data (file upload) multipart/form-data. Using default above.

Can't nest forms (many on same page ok)

Form Submission and Response

User clicks submit button...the Browser...

1. Identifies successful controls
2. Builds a form "data set" with "name=value"-pairs
3. Encode data set depending on content type
4. Sends the data set

User agents should render the response from the HTTP "get" and "post" transactions.

- So it will render a "post"... more to come...(i.e. we will see some page in the browser after the post)!

Get vs Post in Forms

Post is the recommended write operation. Use post in forms

Post

- Data invisible
- Data in message body

Get

- Data appended to URI as a query string (?a=1&b=2...) also hidden data (input type="hidden")! Visible in address field
- Limited URI length, 255 bytes, only accept ASCII
- Possible need URLEncoding ('=' encoded as %3D)

Server Side Form Handling

Possible to extract the incoming form values (=parameters) sent from browser by name (string)

```
// Typical style, getting parameters
// In HTML form <input ... name="key" />
String value = request.getParameter("key");
```

Types for Input Element

```
<input type="..." />
```

For decades just a handful, then came HTML5...
http://www.w3.org/html/wg/drafts/html/master/forms.html#the-input-element

Nice article http://diveintohtml5.info/forms.html

HTML5 also moved formvalidation into the browser (previously had to use JavaScript)

- And also attribute required (i.e. must fill in)

Common input element attributes

`<input.../>`

"These attributes only apply to an input element if its type attribute is in a state whose definition declares that the attribute applies" // :-) HTML 5.1

Attributes

- maxlength, size, readonly, required, pattern, min, max, step, list placeholder, ...

Type "hidden"

```
<input type="hidden" .../>
```

Special. Not for user input

Used by page author to add hidden form parameter

- Useful at server side, some extra information
- Extracted from request like before

Other Form Controls

Quit a few tags

- button, select, datalist, optgroup, option, textarea keygen, output, progress, meter

Form summary: All you need is there, find it.

HTML Events

"DOM … events allow event-driven programming languages like JavaScript, JScript, ECMAScript, VBScript and Java to register various event handlers/listeners on the element nodes inside a DOM tree, e.g. HTML, XHTML, XUL and SVG documents. Historically, like DOM, the event models used by various web browsers had some significant differences. This caused compatibility problems. To combat this, the event model was standardized by the W3C in DOM Level 2." //Wikipedia

More to come, see JQuery...