Topic 2
HTML 5 Structural Elements and Forms
CITS3403 Web & Internet Technologies
HTML5 References and Tutorials

• We will give just a brief overview of common aspects of HTML5. Many references can be found in texts and on-line the official standards see also:
  – W3C Editor’s Draft
  – Books
    – HTML5 for Web Designers, by Jeremy Keith, Published by A Book Apart.
  – HTML5 Tutorials
    – 15 Useful HTML5 Tutorials
    – HTML and the Future of the Web
Philosophies behind HTML5

- Specifying current browser behaviour to ensure interoperability
  - Clear specifications on error handling
- Not breaking the web (backward compatibility)
- Programmatic rather than theoretic (HTML5 is not XML)
- User > Web Designer > Browser Implementer > Standard Theorists
- Aiming at easier authoring of Web Applications
HTML5 is not XML

• For example, all of the below are valid HTML code.

```html
<META CHARSET=UTF-8>
<META CHARSET=UTF-8 />
<META CHARSET="UTF-8">
<META CHARSET="UTF-8" />
<meta charset=utf-8>
<meta charset=utf-8 />
<meTa CHARSET="utf-8">
<meTa CHARSET="utf-8" />
```
Developers’ Word of Caution on HTML5

• Keeping the Web Accessible
  – “As developers we shouldn’t get into pointless discussions of which markup is the best. They all lead to nowhere. Rather, we must get a brand new ideology and modify our coding habits to keep the web accessible.”

• If your site is XHTML strict compliant, no need to change to HTML5.
  – HTML5 is not for everyone. Therefore, you must be wise and select how and where to use it. Think of all the markup flavours you’ve got available as tools: use the right one for the right job. Therefore, if your website is coded in standards compliant XHTML strict there’s no real need to change to HTML5.

Developers’ Word of Caution on HTML5

• Not all features of HTML5 are supported.
  – There’s also the fact that by using HTML5 code right now your website gets stuck in some kind of “limbo” since even though your browser will render HTML5, it does not understand all of it as of yet. This may also apply to other software such as screenreaders and search engines.

• HTML5 is still evolving
  – Lastly you must consider that HTML5 is still under development, and it’s probably the “most open” project the W3C has ever done. With the immense amount of feedback and all the hype around it, the current draft is bound to change and it’s impossible to predict how much.

HTML5 Test and Compare

Graceful Degradation and Progressive Enhancement

- So what are these two terms all about? **Graceful Degradation** is a widely used term which ideology is basically using the latest technologies first, and then fix anything that needs fixing for older browsers. We do this on a daily basis: most of us code for Firefox first, then fix Internet Explorer. That is Graceful Degradation in the practice.

- **Progressive Enhancement** refers to the habit of building first for the less capable, outdated browser and then enhance for the latest technologies. We, too, use this on a daily basis. For example, most of the times we code a website we start with the markup and then apply an external CSS file where we add all the styling. That is Progressive Enhancement in the practice.

- Both technologies usually go hand in hand and have been part of the ways we do things for years. It’s just the terms that are not that well-known. And now, both of these practices need to evolve due to the new languages that are approaching.

So What’s New in HTML 5?

- DOCTYPE

- New Structural Elements more for Designers (taking into consideration both document-like Web pages and no-document-like Web Applications)

- New Canvas Element to support dynamic drawing in browsers

- About 30 new elements like `<time>` to provide more meaningful annotation

- New Form Elements to easy the pain of repeated effort on form validation

- New Global Attributes: contenteditable and designmode
  - [http://www.quackit.com/html_5/tags/html_5_global_attributes.cfm](http://www.quackit.com/html_5/tags/html_5_global_attributes.cfm)
DOCTYPE

• `<!DOCTYPE html>`

• “I was also curious why they chose to ‘html’ rather than ‘html5’, it seemed like the logical way to tell a browser that the current document was written in HTML5, and offered a good template for the future. But I found that `<!DOCTYPE html5>` triggers [Quirks Mode](http://www.wpconfig.com/2009/07/17/html5-and-the-future-of-the-web/) in IE6, and when taking backwards compatibility into consideration `<!DOCTYPE html>` is a pretty good choice (in my opinion).”

• “Overall, I really like the new DOCTYPE; it’s small, meaningful, and maybe we’ll actually be able to remember this one by heart and not have to paste it from site to site.”
Typical Layout in HTML or XHTML

Layout in (X)HTML

Layout in HTML5

In December 2005, Ian Hickson led a web authoring analysis of a sample of slightly over a billion documents, extracting information about popular class names, elements, attributes, and related metadata.


More recently, Opera MAMA crawler also looked at the class attribute in 2.1+ million random URLs and id values of 1.8+ million pages.
SCHOOL OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING

1. `<!doctype html>`
2. `<html>`
3. `<head>`
4. `<title>Page title</title>`
5. `</head>`
6. `<body>`
7. `<header>`
8. `<h1>Page title</h1>`
9. `</header>`
10. `<nav>`
11. `<!-- Navigation -->`
12. `</nav>`
13. `<section id="intro">`
14. `<!-- Introduction -->`
15. `</section>`
16. `<section>`
17. `<!-- Main content area -->`
18. `</section>`
19. `<aside>`
20. `<!-- Sidebar -->`
21. `</aside>`
22. `<footer>`
23. `<!-- Footer -->`
24. `</footer>`
25. `</body>`
26. `</html>`
The new structural elements

- `<header>`
  The header element contains introductory information to a section or page. This can involve anything from our normal documents headers (branding information) to an entire table of contents.

- `<nav>`
  The nav element is reserved for a section of a document that contains links to other pages or links to sections of the same page. Not all link groups need to be contained within the `<nav>` element, just primary navigation.

- `<section>`
  The section element represents a generic document or application section. It acts much the same way a `<div>` does by separating off a portion of the document.
The New Structural Elements (cont.)

- **<article>**
  The article element represents a portion of a page which can stand alone such as: a blog post, a forum entry, user submitted comments or any independent item of content.

- **<aside>**
  Aside, represents content related to the main area of the document. This is usually expressed in sidebars that contain elements like related posts, tag clouds, etc. They can also be used for pull quotes.

- **<footer>**
  The footer element is for marking up the footer of, not only the current page, but each section contained in the page. So, it’s very likely that you’ll be using the <footer> element multiple times within one page.
Document outlining algorithm

It produces outline summaries of Web pages based on how they are marked up.

1. Horses for sale
   1. Mares
      1. Pink Diva
      2. Ring a Rosies
      3. Chelsea’s Fancy
   2. Stallions
      1. Korah’s Fury
      2. Sea Pioneer
      3. Brown Biscuit

Online Outliner Testing Tool: http://gsnedders.html5.org/outliner/
What elements affects outline?

• Two things in the mark-up affect the outline of a Web page:

  – Implicit: heading content (h1 to h6 and hgroup)
    • The heading elements have a rank
      – <h1> has the highest rank, and
      – <h6> has the lowest rank.
  – Explicit: sectioning content (section, article, aside and nav).
    • Explicit outline also needs headings to be useful.
    • The basic rule is: the first HTML heading element (one of <h1>, <h2>, <h3>, <h4>, <h5>, <h6>) defines the heading of the current section.
  – Relative ranking matters only within a section; the structure of the sections determines the outline, not the heading rank of the sections.

1. Horses for sale
   1. Mares
      1. Pink Diva
      2. Ring a Rosies
      3. Chelsea’s Fancy
   2. Stallions
      1. Korah's Fury
      2. Sea Pioneer
      3. Brown Biscuit

<p>All our horses come with full paperwork and a family tree.</p>
HTML5 Forms

• New Attributes
  – placeholder
  – autocomplete (on, off)
  – required
  – autofocus

• New form controls
  – datalist

• New input types
  – search
  – Contacts (email, url, tel)
  – Slider: range
  – Spinner: number
  – Data/time: date, datetime, datetime-local, month, week

• Simplicity
• Consistency
• Validation
• Frequent design patterns
### HTML5 Forms - Summary

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type=&quot;text&quot;</td>
<td>Text field</td>
</tr>
<tr>
<td>type=&quot;radio&quot;</td>
<td>Radio button</td>
</tr>
<tr>
<td>type=&quot;check&quot;</td>
<td>Check box</td>
</tr>
<tr>
<td>type=&quot;submit&quot;</td>
<td>Submit button</td>
</tr>
<tr>
<td>type=&quot;reset&quot;</td>
<td>Reset button</td>
</tr>
<tr>
<td>type=&quot;file&quot;</td>
<td>File input</td>
</tr>
<tr>
<td>type=&quot;email&quot;</td>
<td>Email input</td>
</tr>
<tr>
<td>type=&quot;tel&quot;</td>
<td>Telephone input</td>
</tr>
<tr>
<td>type=&quot;url&quot;</td>
<td>URL input</td>
</tr>
<tr>
<td>type=&quot;date&quot;</td>
<td>Date input</td>
</tr>
<tr>
<td>type=&quot;search&quot;</td>
<td>Search input</td>
</tr>
<tr>
<td>type=&quot;range&quot;</td>
<td>Range input</td>
</tr>
<tr>
<td>type=&quot;number&quot;</td>
<td>Number input</td>
</tr>
<tr>
<td>type=&quot;color&quot;</td>
<td>Color picker</td>
</tr>
<tr>
<td>type=&quot;date</td>
<td>datetime</td>
</tr>
</tbody>
</table>

```html
<textarea type="text"><output>
</textarea>
```
HTML5 Form Demo

• My attempt

• Only supported by Opera 9.5+
  – http://www.brucelawson.co.uk/tests/html5-forms-demo.html
The Mobile Safari changes on-screen keyboard according to different contact input types:
HTML 5 Forms: Dates and times

- One of the most popular JavaScript widgets is the calendar picker.

- These calendar widgets all do the same thing, but you’ll find that they’re implemented slightly differently on each site. A native calendar widget would smooth away the inconsistencies and reduce cognitive load during the date-picking process.

- HTML 5 introduces a raft of input types specifically for dates and times
  - **date** is for a year, month, and day.
  - **datetime** is for a year, month, and day in combination with hours, minutes, and seconds and time zone information.
  - **datetime-local** is the same but without the time zone information.
  - **time** is for hours, minutes, and seconds.
  - **month** is for a year and a month but without a day.
HTML 5 Forms: Dates and times (Cont.)

- All of these input types will record timestamps with some subset of the standardized format YYYY-MM-DDThh:mm:ss.Z (Y is year, M is month, D is day, h is hour, m is minute, s is second, and Z is timezone). Take, for example, the date and time at which World War One ended, 11:11am on November 11th, 1918:
  - date: 1918-11-11
  - datetime: 1918-11-11T11:11:00+01
  - datetime-local: 1918-11-11T11:11:00
  - time: 11:11:00
  - month: 1918-11

- There is no year input type, although there is a week input type that takes a number between 1 and 53 in combination with a year.
HTML 5 Forms: color

- `<input type="color">` produce a color picker.
HTML 5 Forms – Build-in Validation

• Native validation without scripting.

• HTML5 browsers support basic validation on email, url and tel input types.

• HTML5 has made it even more friendly for web authoring
  – The pattern attribute that allows you to use regular expression to specify required format
  – For example:

    <input id="phone" name="phone" pattern="\d{8}" type="tel">
The form output element

- Represent the results of some calculation

Example:

```html
<form oninput="result.value=parseInt(a.value)+parseInt(b.value)">
0<input type="range" name="b" value="50" />100 +
<input type="number" name="a" value="10" />
<output name="result"></output>
</form>
```
The `<time>` element

- Unambiguously encode time and date in formats that are both
  - Machine readable, and
  - Human readable

```xml
<time datetime="2011-8-12">12 August 2011</time>
<time datetime="2011-8-12">12 <sup>th</sup> August Last Year</time>
<time datetime="2012-8-12">UWA Expo 2012</time>
<time datetime="2012-8-12T14:00Z">2PM on UWA Expo 2012</time>
<time datetime="20:00">8PM</time>
```
The `<time>` element - Fuzzy Dates

- Previously, you could only mark up precise dates, which could be a problem (e.g. historians)
  - 13 November 1905 could be expressed in HTML but not November 1905
    `<time datetime="1905-11-13">`
- Now, "fuzzy dates" are possible:
  - `<time datetime="1905">` means the year 1905
  - `<time datetime="1905-11">` means November 1905
  - `<time datetime="11-13">` means 13 November (any year)
  - `<time datetime="1905-W21">` means week 21 of 1905
The `<time>` element - timezone

- As before, times are expressed using the 24 hour clock.

- Now, you can separate the date and time with a space rather than a "T" (but you don't have to). So both of these are valid:

  `<time datetime="1905-11-13T09:00">`
  `<time datetime="1905-11-13 09:00">`

- You can localise times, as before. Appending "Z" to the timezone indicates UTC (a way of saying "GMT" without it being comprehensible to normal people). Otherwise, use an offset:

  `<time datetime="09:00Z">` is 9am, UTC.
  `<time datetime="09:00-05:00">` is 9am in the timezone 5 hours behind UTC.
  `<time datetime="09:00+05:45">` is 9am in Nepal, which is UTC + 5 hours and 45 minutes.
The `<time>` element - Durations

- In New! Improved! HTML5 `<time>`, you can represent durations, with the prefix "P" for "period".
  - The datetime attribute value: "D" for days, "H" for hours, "M" for minutes and "S" for seconds.
- You can separate them with spaces (but you don't have to).
  - `<time datetime="P4D">` is a duration of 4 days, same as
  - `<time datetime="P 4 D">`
- Using a "T" after the "P" marker allows you to be more precise:
  - `<time datetime="PT23H 9M 2.345S">` is a duration of 23 hours, 9 minutes and 2.345 seconds.
- Alternatively, you can use a duration time component.

http://introducinghtml5.com/errata/ch02.html
The pubdate attribute in `<time>`

- A Boolean attribute used to indicate the publication datetime.

```xml
<section>
  <article>
    <header>
      <h1>Seminar: What is ARIA?</h1>
      <p><time datetime="2012-08-12T11:00">12 August 2012 11:00am</time></p>
    </header>
    <p>This seminar is about accessibility.</p>
    <footer>
      Published at: <time datetime="2012-08-08T20:00" pubdate>8 August 2012 8:00pm</time>
    </footer>
  </article>
</section>
```
The `<mark>` element

- Works as a web page high-lighter.

 `<p>`This seminar is about `<mark>`accessibility`</mark>`.</p>`

This seminar is about accessibility.
Figure and Figure Captions

- `<figure>` element represents some self-contained content, usually this is an image, an illustration, a diagram, a code snippet, or a schema that is referenced in the main text.

- It can be moved to another page or to an appendix without affecting the main flow.

- Frequently used together with `<figurecaption>`.

```html
<figure>
  <img alt="An awesome picture" src="picture.jpg">
  <figcaption>Caption for the awesome</figcaption>
</figure>
```
The **audio Element**

- Prior to HTML5, a plug-in was required to play sound while a document was being displayed

- Audio encoding algorithms are called *audio codecs* – e.g., MP3, Vorbis

- Coded audio data is stored in containers—e.g., Ogg, MP3, and Wav (file name extension indicates the container, not the audio code)
  - Vorbis code is stored in Ogg containers
  - MP3 code is stored in MP3 container
  - Wav code is stored in Wav containers
The audio Element (continued)

- General syntax:

```html
<audio attributes>
    <source src = "filename_1" >
    ...
    <source src = "filename_n" >
Your browser does not support the audio element
</audio>
```

- The controls attribute, which is set to `controls`, creates a start/stop button, a clock, a progress slider, total time of the file, and a volume slider.
The **audio** Element (continued)

- Browser chooses the first audio file it can play and skips the content; if none, it displays the content.
- Different browsers have different audio capabilities:
  - Firefox 3.5+ supports Ogg/Vorbis and Wav/Wav
  - Chrome 3.0+ supports Ogg/Vorbis and MP3/MP3
  - IE9 supports MP3/MP3
  - Safari 3.0+ supports Wav/Wav

<table>
<thead>
<tr>
<th></th>
<th>.OGG/ .OGV (VORBIS CODEC)</th>
<th>MP3</th>
<th>MP4/ M4A (AAC CODEC)</th>
<th>WAV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opera</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Firefox</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Chrome</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>IE9</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Safari</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
The audio Element (continued)

<!DOCTYPE html>
<!– audio.html Test the audio element -->
<html lang = "en" >
<head>
<title> Test audio element </title>
<meta charset = "utf-8" />
</head>
<body>
This is a test of the audio element
<audio controls = "controls" >
<source src = "nineoneone.ogg" />
<source src = "nineoneone.wav" />
<source src = "nineoneone.mp3" />
Your browser does not support the audio element
</audio>
</body>
</html>
The video Element

• Prior to HTML5, there was no standard way to play video clips while a document was being displayed

```html
<video width="600" height="500" autoplay="autoplay"
controls="controls" preload="preload">
<source src="" />
<source src="" />
...
Your browser does not support the video element
</video>
```

• The width and height attributes set the screen size

• The autoplay attribute, set to "autoplay", specifies that the video should play as soon as it is ready

• The preload attribute, set to "preload", specifies that the video should be loaded as soon as the document is loaded.
The video Element (continued)

<!DOCTYPE html>
<!-- testvideo.html test the video element -->
<html lang = "en">
<head>
  <meta charset = "UTF-8" />
  <title> test video element </title>
</head>
<body>
  This is a test of the video element.....
  <video width = "600" height = "500" autoplay = "autoplay"
        controls = "controls" preload = "preload">
    <source src = "NorskTippingKebab.mp4" />
    <source src = "NorskTippingKebab.ogv" />
    <source src = "NorskTippingKebab.webm" />
  Your browser does not support the video element
</video>
</body>
</html>
The video Element (continued)

• Video codecs:
  – H.264 (MPEG-4 AVC) – can be stored in an MPEG-4 container
  – Theora – can be stored in any container
  – VP8—can be stored in WebM container

• Video codecs are stored in containers
  – Common containers MPEG-4 (.mp4), Flash Video (.flv), Ogg (.ogv), WebM (.webm) and Audio Video Interleave (.avi).
  – Convert video format free online: http://video.online-convert.com/
Browser Video Codec Support

- Different browsers natively support different codecs
  - Firefox 3.5+ supports Ogg
  - Firefox 4.0+ supports Ogg and WebM
  - Chrome 6.0+ supports all three most common containers
  - IE9 supports MPEG-4
  - Safari 3.0+ supports MPEG-4

<table>
<thead>
<tr>
<th></th>
<th>WEBM (VP8 Codec)</th>
<th>MP4 (H.264 Codec)</th>
<th>OGV (OGG THEORA Codec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opera</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Firefox</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Chrome</td>
<td>Yes</td>
<td>Yes—see Note, support will be discontinued</td>
<td>Yes</td>
</tr>
<tr>
<td>IE9 +</td>
<td>Yes (but codec must be installed manually)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Safari</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
HTML 5 Canvas Demo

- http://www.blobsallad.se/
Offline Data Storage

• The coolest part about HTML5 is definitely its offline capabilities. Programs like Thunderbird and Outlook (and now GMail to an extent) let you browse through your old data while staying offline. With HTML5, you’ll have this same functionality, but in the browser. This is the first serious step towards bridging the gap between the desktop and the Web, and opens all sorts of doors for the future of Web apps.

• The W3C has taken the best parts from the various Web technologies and rolled them into, what is being dubbed the most powerful markup language to date.

Some other features of the HTML5 APIs

- **Drag & Drop**
  The drag and drop API defines an event-based drag and drop system. However, it never defines what "drag and drop" is. This API requires JavaScript to fully work as normal think drag and drop functionality.

- **Geolocation**
  Geolocation is a very cool API available within HTML5. Its object can be used to programmatically determine location information through a device’s user agent (hint: mobile devices).
Useful Cheat Sheet

• HTML5 Quick Reference
  

• HTML5 and HTML4 Comparison at a Glance
  

• HTML5 Canvas
  