



ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

# IT4409: Web Technologies and e-Services

## Lec 2: HTML

# Objectives

- ❖ Describe hypertext and HTML standards
- ❖ Understand HTML elements and markup tags
- ❖ Create the basic structure of an HTML file
- ❖ Insert an HTML comment
- ❖ Work with block-level elements
- ❖ Create lists, tables, hyperlinks and insert images
- ❖ Learn HTML5 tags
- ❖ Work with forms and inputs

# Outline

## 1. Basic HTML

- hypertext
- tags & elements
- text formatting
- lists, hyperlinks, images
- tables, frames

## 2. Advanced HTML

# Hypertext & HTML

- HyperText Markup Language (HTML) is the language for specifying the *static* content of Web pages (based on SGML, the Standard Generalized Markup Language)
  - *hypertext* refers to the fact that Web pages are more than just text
    - can contain multimedia, provide links for jumping within the same document & to other documents
  - *markup* refers to the fact that it works by augmenting text with special symbols (tags) that identify the document structure and content type

# Hypertext & HTML (cont.)

- HTML is an evolving standard (as new technology/tools are added)
  - HTML 1 (Berners-Lee, 1989): very basic, limited integration of multimedia  
in 1993, Mosaic added many new features (e.g., integrated images)
  - HTML 2.0 (IETF, 1994): tried to standardize these & other features, but late  
in 1994-96, Netscape & IE added many new, divergent features
  - HTML 3.2 (W3C, 1996): attempted to unify into a single standard  
but didn't address newer technologies like Java applets & streaming video
  - HTML 4.0 (W3C, 1997): current standard (but moving towards XHTML)  
attempted to map out future directions for HTML, not just react to vendors
  - XHTML 1.0 (W3C, 2000): HTML 4.01 modified to conform to XML standards
  - XHTML 1.1 (W3C, 2001): “Modularization” of XHTML 1.0
  - HTML 5 (Web Hypertext Application Technology Working Group, W3C, 2006):  
New  
version of HTML4, XHTML 1.0, and DOM 2 (still a work in progress), no longer based on  
SGML, but “backward compatible” with parsing of older versions of HTML

# Web Development Tools

- Many high-level tools exist for creating Web pages  
e.g., Microsoft FrontPage, Netscape Composer, Adobe PageMill,  
Macromedia DreamWeaver, HotDog, ...  
also, many applications have "save to HTML" options (e.g., Word)

## Don't use these tools!!

*for most users who want to develop basic, static Web pages, these are fine (but many of these programs produce very poorly structured HTML code)*

why are we learning low-level HTML using a basic text editor?

- may want low-level control
- may care about size/readability of pages
- may want to "steal" components from other pages and integrate into existing pages
- may want dynamic features such as scripts or applets
- remote editing of web pages may only be possible using a basic text editor
- **sticking to (internationally and industrially) agreed upon standards will help ensure your web documents are rendered as you intend them to look and operate as you desire**

# Tags and Elements

- HTML specifies a set of *tags* that identify structure of the document and the content type
  - tags are enclosed in `< >`
    - `` specifies an image
  - most tags come in pairs, marking a beginning and ending
    - `<title>` and `</title>` enclose the title of a page
- An HTML *element* is an object enclosed by a pair (in most cases) of tags
  - `<title>My Home Page</title>` is a TITLE element
  - `<b>This text appears bold.</b>` is a BOLD element
  - `<p>Part of this text is <b>bold</b>. </p>` is a PARAGRAPH element that contains a BOLD element

*An HTML document is a collection of elements (text/media with context).*

# Structural Elements

- an HTML document has two main structural elements
  - **HEAD** contains setup information for the browser & the Web page
    - e.g., the title for the browser window, style definitions, JavaScript code, ...
  - **BODY** contains the actual content to be displayed in the Web page

```
<html>
  <!-- Version information --
  -- File: page01.html --
  -- Author: CS443--
  -- Creation: 22.09.09 --
  -- Description: introductory page --
  -- Copyright: U.Liverpool --
  -->
  <head>
    <title>My first HTML document</title>
  </head>
  <body>
    <p> Hello world! </p>
  </body>
</html>
```

HTML documents begin and end with `<html>` and `</html>` tags

Comments appear between `<!--` and `-->`

**HEAD** section enclosed between `<head>` and `</head>` tags

**BODY** section enclosed between `<body>` and `</body>`

*\* Find more info on HTML docs!*



# <head> and <body> elements

- The **<head>** element is where you include a **<title>** element (that appears in the title bar of the browser).
- You can also include lots of other type of information in the **<head>** element.
  - Cascading Style sheet information, or a link to an external style sheet (or several)
  - “Meta” data, such as who authored the page, the type of content, and clues that search engines may (or may not) use to help categorize your page
  - JavaScript code
- The **<body>** element contains the main bulk of the material to be displayed on the webpage.
  - Paragraphs
  - Tables and lists
  - Images
  - JavaScript code
  - PHP code can be included here too (if passed through a PHP parser before being served to the client’s browser)
  - Other embedded objects

# Text Layout

```
<html>
<!-- CS443 page02.html 17.09.14 -->
<head>
  <title>Text Layout</title>
</head>

<body>
  <p>
    This is a paragraph of text<br/>
    made up of two lines.
  </p>

  <p>
    This is another paragraph with a
    &nbsp; GAP &nbsp; between
    some of the words.
  </p>

  <p>
    &nbsp;&nbsp;&nbsp; This paragraph is<br/>
    indented on the first line<br/>
    but not on subsequent lines.
  </p>
</body>

</html>
```

[view page](#)

❖ for the most part, layout of the text is left to the browser

- (almost) every sequence of whitespace is interpreted as a single space
- browser automatically wraps the text to fit the window size

❖ can override some text layout

- can specify a new paragraph (starts on a new line, preceded by a blank line) using `<p>...</p>`
- can cause a line break using the `<br/>` tag (“self-closing” tag)
- can force a space character using the symbol for a “non-breaking space”: `&nbsp;`

# Separating Blocks of Text

```
<html>
<!-- CS443 page03.html 15/08/06 -->
<head>
  <title>Blocks of Text</title>
</head>

<body>
  <h1>Major heading 1</h1>
  <p>
    Here is some text.
  </p>

  <h2>Subheading</h2>
  <p>
    Here is some subtext.
  </p>

  <hr/>

  <h1>Major heading 2</h1>
  <p>
    Here is some more text.
  </p>
</body>

</html>
```

[view page](#)

❖ can specify headings for paragraphs or blocks of text

- `<h1>...</h1>` tags produce a large, bold heading
- `<h2>...</h2>` tags produce a slightly smaller heading
- ...
- `<h6>...</h6>` tags produce a tiny heading

❖ can insert a horizontal rule to divide sections

- `<hr/>` draws line across window

# The Basic Web page – A Worked Example

```
<html>
<!-- CS443  page22.html  17.10.14  -->
  <head>
    <title> Bill Smiggins Inc. </title>
  </head>
  <body>
    <h1>Bill Smiggins Inc.</h1>
    <h2>About our Company...</h2>
    <p>This Web site provides clients, customers,
      interested parties and our staff with all of
      the information that they could want on
      our products, services, success and failures.
    </p>
    <hr/>
    <h3> Products </h3>
    <p> We are probably the largest
      supplier of custom widgets, thingummybobs, and bits
      and pieces in North America. </p>
    <hr/>
  </body>
</html>
```

# Text Appearance

```
<html>
  <!-- CS443 page25.html 15.08.06 -->
  <head>
    <title>Text Variations and Escape
    Sequences</title>
  </head>
  <body>
    <h1>Text Variations</h1>
    <p>We can use <b>simple</b> tags to
      <i>change</i> the appearance of
      <strong>text</strong> within
      <tt>Web pages</tt>.
      Even super<sup>script</sup>
      and sub<sub>scripts</sub> are
      <em>supported</em>.</p>

    <h1>Text Escape Sequences</h1>
    <p>
      &amp; &lt; &gt; &quot; &copy;
    </p>
    <h1>Preformatted text</h1>
    <pre>
      University of Liverpool
      Department of Computer Science
      Ashton Building, Ashton Street
      Liverpool, L69 3BX, UK
    </pre>
  </body>
</html>
```

## ❖ can specify styles for fonts

- `<b>... </b>` specify bold
- `<i>... </i>` specify italics
- `<tt>... </tt>` specify typewriter-like (fixed-width) font
- `<big>... </big>` increase the size of the font
- `<small>... </small>` decrease the size of the font
- `<em>... </em>` put emphasis
- `<strong>... </strong>` put even more emphasis
- `<sub>... </sub>` specify a subscript
- `<sup>... </sup>` a superscript
- `<pre>... </pre>` include ready-formatted text
- `&amp; &al; &gt; &quot; &copy;` escape characters used in HTML control
- *Find more info on text tags!*

# Lists

```
<html>
<!-- CS443page07.html 23.09.08 -->
<head> <title>(Sort of) Simple Lists</title>
  <style type="text/css">
    .my_li:before { content: counter(list) ": ";
                  counter-increment: list; }
  </style> </head>
<body>

<ul style="list-style-type: square;">
  <li> ... first list item... </li>
  <li> ... second list item... ... </li>
</ul>
<dl>  <dt> Dweeb </dt>
      <dd> young excitable person who may
      mature into a <em>Nerd</em> </dd>
  <dt> Hacker </dt>
      <dd> a clever programmer </dd>
  <dt> Nerd </dt>  <dd> technically bright but
      socially inept person </dd>
</dl>
<ol style="list-style-type: none;
        counter-reset: list 29;" >
  <li class="my_li">Makes first item number 30.</li>
  <li class="my_li">Next item continues to number
31.</li>
</ol>

</body>
</html>
```

## ❖ there are 3 different types of list elements

- `<ol>...</ol>` specifies an ordered list (using numbers or letters to label each list item)
  - `<li>` identifies each list item
  - *can set type of ordering, start index*
- `<ul>...</ul>` specifies unordered list (using a bullet for each)
  - `<li>` identifies each list item
- `<dl>...</dl>` specifies a definition list
  - `<dt>` identifies each term
  - `<dd>` identifies its definition

\* We will learn more about the “style” attributes soon enough.

# Hyperlinks

```
<html>
<!-- CS443page08.html 17.10.14 -->

<head>
  <title>Hyperlinks</title>
</head>

<body>
  <p>
    <a href="http://www.liv.ac.uk">
      The University of Liverpool</a>
    <br/>
    <a href="page07.html"
      target="_blank">
      Open page07 in a new window</a>
    </p>
  </body>

</html>
```

view page

❖ perhaps the most important HTML element is the hyperlink, or ANCHOR

- `<a href="URL">...</a>`
  - where URL is the Web address of the page to be displayed when the user clicks on the link
  - *if the page is accessed over the Web, must start with http://*
  - *if not there, the browser will assume it is the name of a local file*
- `<a href="URL" target="_blank">...</a>`
  - causes the page to be loaded in a new Window

\* Find more info on attribute TARGET

# Hyperlinks (cont.)

```
<html>
<!-- CS443 page09.html 21.09.12 -->

<head>
  <title>Internal Links in a Page</title>
</head>

<body>
  <p>
    [ <a href="#HTML">HTML</a> |
      <a href="#HTTP">HTTP</a> |
      <a href="#IP">IP</a> |
      <a href="#TCP">TCP</a> ]
  </p>
  <p>
    Computer acronyms:
    <dl>
      <dt id="HTML">HTML</dt>
      <dd>HyperText Markup Language
      <dt id="HTTP">HTTP</dt>
      <dd>HyperText Transfer Protocol...</dd>
      <dt id="IP">IP</dt>
      <dd>Internet Protocol...</dd>
      <dt id="TCP">TCP</dt>
      <dd>Transfer Control Protocol...</dd>
    </dl>
  </p>
</body>
</html>
```

❖ for long documents, you can even have links to other locations in that same document

- `<xxxx id="ident">...</xxxx>`
  - where `ident` is a variable for identifying this location, where "xxxx" can, in principle, be any HTML element
  - (this is actually an HTML5 language specification, but seems to work in most browsers)
- `<a href="#ident">...</a>`
  - will then jump to that location within the file
- `<a href="URL#ident">...</a>`
  - can jump into the middle of another file just as easily



# Images

## ❖ can include images using img

- by default, browsers can display GIF and JPEG files, more modern browsers can also typically support PNG files and SVG graphics (of course, use at your own risk)
- other image formats may require plug-in applications for display
  - ``
  - again, if file is to be accessed over the Web, must start with `http://` (if not, will assume local file)
  - Find more info on `<img/>`

```
<html>
<!-- CS443 page10.html 18.09.13 -->
<head>
  <title>Image example</title>
</head>
<body>


<p>The Anglican Cathedral of Liverpool</p> </body>
</html>
```

view page

# Images (cont.)

- **src** - specifies the file name (and can include a URL)
- **width** and/or **height** - dimensions in pixels (often only need to specify one of them and the other is automatically scaled to match, where possible pictures should be resized using other programs to save on bandwidth and problems that some (older) browsers might have with resizing images)
- **title** - displayed when the mouse is “hovered” over the picture
- **alt** - text that is displayed when the image is missing, can’t be loaded (e.g. if file permissions aren’t set correctly), or if the client has disabled loading images in his/her browser

# Tables

- tables are common tools for arranging complex layout on a Web page
  - a table divides contents into rows and columns
  - by default, column entries are left-justified, so you must provide for your own alignment when needed (using Cascading Style Sheets, for example)

```
<html>
<!-- CS443 page11.html 17.10.14 -->
<head>
  <title>Tables</title>
</head>
<body>
<h2>A Simple Table</h2>
  <table>
    <tr>
      <td> Left Column </td>
      <td> Right Column </td>
    </tr>
    <tr>
      <td> Some data </td>
      <td> Some other data </td>
    </tr>
  </table>
</body>
</html>
```

`<table>...</table>` specify a table element

`<tr>...</tr>` specify a row in the table

`<td>...</td>` specify table data (i.e., each column entry in the table)

[view page](#)

# Layout in a Table

```
<html>
<!-- CS443 page12.html 17.10.14 -->

<head>
  <title>Table Layout</title>
</head>

<body>
  <table style="border: 1px solid;">
    <tr style="text-align: center;">
      <td style="border: 1px solid;">
        Left<br/>Column</td>
      <td style="border: 1px solid;
        vertical-align: top;">
        Right Column</td>
    </tr>
    <tr>
      <td style="border: 1px solid;">
        Some data</td>
      <td style="border: 1px solid;">
        Some data</td>
    </tr>
  </table>
</body>
</html>
```

[view page](#)

❖ can have a border on tables using the “style” attribute

```
<table style= "border: 1px
solid;">
```

*increasing the number makes the border thicker*

❖ can control the horizontal & vertical layout within cells

```
<td style= "text-align:center">
<td style= "vertical-align: bottom">
```

❖ can apply layout to an entire row

```
<tr style="text-align: center">
<tr style="vertical-align: top">
```

We will explore this more with Cascading Style Sheets (CSS).

# Table Width

```
<html>
<!-- CS443 page13.html 17.10.14 -->
<head>
  <title>Table Width</title>
</head>

<body>
<table style="width: 100%;">
  <tr>
    <td>left-most </td>
    <td style="text-align: right;">
      right-most</td>
  </tr>
</table>
</body>
</html>
```

[view page](#)

❖ by default, the table is sized to fit the data

❖ can override & specify the width of a table relative to the page

For example

```
<table style="width:
60%">
```

# Other Table Attributes

```
<html>
<!-- CS443 page14.html 17.10.14 -->
<head>
  <title>Table Formatting</title>

  <style type="text/css" media="screen">
    table { border: 1px solid; padding: 1px; }
    th, td { border: 1px solid; padding: 10px;
              text-align: center; }
  </style>
</head>

<body>
  <table>
    <tr>
      <th>HEAD1</th> <th>HEAD2</th>
<th>HEAD3</th>
    </tr>
    <tr>
      <td>one</td> <td>two</td> <td>three</td>
    </tr>
    <tr>
      <td rowspan="2"> four </td>
      <td colspan="2"> five </td>
    </tr>
    <tr>
      <td> six </td> <td> seven </td>
    </tr>
  </table>
</body>
</html>
```

view page

❖ can control the space between cells & margins within cells

This is the “padding” attribute in the table and th,td style sheet declarations (more on this with Cascading Style Sheets).

❖ can add headings

`<th>` is similar to `<td>` but displays heading centered in bold

❖ can have data that spans more than one column

`<td colspan="2">`

❖ similarly, can span more than one row

`<td rowspan="2">`

(This example uses CSS style sheet commands in the page <header>.)

# Frames

- Frames provide the ability to split the screen into independent parts
- ❖ Frames are going out of fashion, partly because they interact poorly with web search engines (i.e. search engines cannot generally access the data stored in the inset frame objects).
- ❖ Frames can also “break” the regular behaviour of browsers, most notably the “Back” button on the browser can behave in unexpected ways.
- ❖ Because of these drawbacks to frames, I will not be discussing them in this course.
- ❖ If you wish to design websites using frames (why would you??), I would encourage you to use the XHTML XFrames specifications (see the W3C website for more details), but this specification isn’t fully supported by all browsers at this time.
- ❖ Frames are also not supported by the HTML 5 specification.

# Content vs. Presentation

- Most HTML tags define content type, independent of presentation.
  - exceptions? (e.g. `<b> ..... </b>` for bold text and `<i> ..... </i>` for italicized text)
- Style sheets associate presentation formats with HTML elements.
  - CSS1: developed in 1996 by W3C
  - CSS2: released in 1998, but still not fully supported by all browsers
  - CSS3: specification still under development by the W3C, “completely backwards compatible with CSS2” (according to the W3C)
- The trend has been towards an increasing separation of the content of webpages from the presentation of them.
- Style sheets allow us to maintain this separation, which allows for easier maintenance of webpages, and for a consistent look across a collection of webpages.



# Content vs. Presentation (cont.)

- ❖ Style sheets can be used to specify how tables should be rendered, how lists should be presented, what colors should be used on the webpage, what fonts should be used and how big/small they are, etc.
- ❖ HTML style sheets are known as *Cascading Style Sheets*, since can be defined at three different levels
  1. *inline* style sheets apply to the content of a single HTML element
  2. *document* style sheets apply to the whole BODY of a document
  3. *external* style sheets can be linked and applied to numerous documents, might also specify how things should be presented on screen or in print *lower-level style sheets can override higher-level style sheets*
- ❖ User-defined style sheets can also be used to override the specifications of the webpage designer. These might be used, say, to make text larger (e.g. for visually-impaired users).

# Inline Style Sheets

```
<html>
<!-- CS443 page17.html 17.10.14 -->

<head>
  <title>Inline Style Sheets</title>
</head>

<body>
  <p style="font-family:Arial,sans-
serif;
           text-align:right">This is a
right-justified paragraph in a sans
serif
font (preferably Arial), with some
  <span style="color:green">green
text</span>.
  </p>

  <p>And <a style="color:red;
                text-decoration:none;
                font-size:larger;"
            href="page01.html">here</a>
    is a formatted link.
  </p>
</body>
</html>
```

view page

❖ Using the `style` attribute, you can specify presentation style for a single HTML element

- within tag, list sequence of **property:value** pairs separated by semi-colons

`font-family:Courier,monospace`

`font-style:italic`

`font-weight:bold`

`font-size:12pt font-size:large font-size:larger`

`color:red color:#000080`

`background-color:white`

`text-decoration:underline`

`text-decoration:none`

`text-align:left text-align:center`

`text-align:right text-align:justify`

`vertical-align:top vertical-align:middle`

`vertical-align:bottom`

`text-indent:5em text-indent:0.2in`

# Inline Style Sheets (cont.)

```
<html>
<!-- CS443 page18.html 17.09.09 -->

<head>
  <title>Inline Style Sheets</title>
</head>

<body>
  <p>Here is an image
    
    embedded in text.
  </p>

  <ol style="list-style-type:upper-alpha">
    <li> one thing</li>
    <li> or another</li>
    <ul style="list-style-type:square;
              whitespace:pre">
      <li> with this</li>
      <li> or that</li>
    </ul>
  </ol>
</body>
</html>
```

## ❖ more style properties & values

margin-left:0.1in      margin-right:5%

margin:3em

padding-top:0.1in      padding-  
                  bottom:5%

padding:3em

border-width:thin      border-width:thick

border-width:5

border-color:red

border-style:dashed      border-style:dotted

border-style:double      border-style:none

whitespace:pre

list-style-type:square

list-style-type:decimal

list-style-type:lower-alpha

list-style-type:upper-roman

# Inline Style Sheets (cont.)

```
<html>
<!-- CS443 page19.html 17.10.14 -->

<head>
  <title> Inline Style Sheets </title>
</head>

<body>
  <table style="font-family:Arial,sans-serif">
    <caption style="color:red;
                  font-style:italic;
                  text-decoration:underline">
      Student data. </caption>
    <tr style="background-color:red">
      <th> name </th> <th> age </th>
    </tr>
    <tr>
      <td> Chris Smith </td> <td> 19 </td>
    </tr>
    <tr>
      <td> Pat Jones </td> <td> 20 </td>
    </tr>
    <tr>
      <td> Doogie Howser </td> <td> 9 </td>
    </tr>
  </table>
</body>
</html>
```

[view page](#)

❖ style sheets can be applied to tables for interesting effects

# Document Style Sheets

- Inline style sheets apply to individual elements in the page.
  - using inline style directives can lead to inconsistencies, as similar elements are formatted differently
    - e.g., we might like for all `<h1>` elements to be centered
  - inline definitions mix content & presentation
    - ➔ violates the general philosophy of HTML
- As a general rule, inline style sheet directives should be used as sparingly as possible
- Alternatively, document style sheets allow for a cleaner separation of content and presentation.
  - style definitions are placed in the `<head>` of the page (within `STYLE` tags)
  - can apply to all elements, or a subclass of elements, throughout the page

# Document Style Sheets

```
<html>
<!-- CS443 page20.html 17.10.14 -->

<head>
  <title>Document Style Sheets</title>
  <style type="text/css">
    h1 {color:blue;
        text-align:center}
    p.indented {text-indent:0.2in}
  </style>
</head>

<body>
  <h1> Centered Title </h1>

  <p class="indented">This paragraph
will have the first line indented, but
subsequent lines will be flush. </p>

  <p>This paragraph will not be
indented.
  </p>

  <h1> The End </h1>

</body>
</html>
```

- ❖ document style sheets ensure that similar elements are formatted similarly
  - can even define **subclasses** of elements and specify formatting

`p.indented` defines subclass of paragraphs

- inherits all defaults of `<p>`
- adds new features

to specify this newly defined class, place `class="ID"` attribute in tag

- ❖ note how "clean" the `<body>` element is

[view page](#)

# Document Style Sheets (cont.)

```
<html>
<!-- CS443 page21.html 17.10.14 -->

<head>
  <title> Inline Style Sheets </title>
  <style type="text/css">
    table {font-family:Arial,sans-serif}
    caption {color:red;
              font-style:italic;
              text-decoration:underline}
    th {background-color:red}
  </style>
</head>

<body>
  <table>
    <caption> Student data. </caption>
    <tr><th> name </th>          <th> age</th></tr>
    <tr><td> Chris Smith </td>    <td> 19 </td></tr>
    <tr><td> Pat Jones </td>      <td> 20 </td></tr>
    <tr><td> Doogie Howser </td> <td> 9 </td></tr>
  </table>
</body>
</html>
```

view page

- ❖ document style sheets are especially useful in formatting tables
- ❖ effectively separates content from presentation
- ❖ what if you wanted to right-justify the column of numbers?
- ❖ what if you changed your mind?

# Pseudo-Elements

```
<html>
<!-- CS443 page23.html 17.10.14 -->

<head>
  <title>Title for Page</title>
  <style type="text/css">
    a {color : red;
      text-decoration : none;
      font-size : larger}
    a:visited {color : black}
    a:active {color : orange}
    a:hover {color : blue}
    p:first-letter {font-size : large;
      color : white;
      background-color : darkblue}
  </style>
</head>

<body>
  <p> Welcome to my Web page. I am so
  happy you are here.
  </p>
  <p> Be sure to visit
  <a href="http://www.cnn.com">CNN</a>
  for late-breaking news.
  </p>
</body>
</html>
```

view page

❖ pseudo-elements are used to address sub-parts of elements

- can specify appearance of link in various states  
•:visited      :active      :hover
- can specify format of first line in page or paragraph  
•:first-line
- can specify format of first letter in page or paragraph  
•:first-letter

❖ *Danger* : changing the look of familiar elements is confusing

❖ *Careful* : current browsers do not support all CSS2 features



# External Style Sheets

- modularity is key to the development and reuse of software
  - design/implement/test useful routines and classes
  - package and make available for reuse
  - saves in development cost & time
  - central libraries make it possible to make a single change and propagate the changes
- external style sheets place the style definitions in separate files
  - multiple pages can link to the same style sheet, consistent look across a site
  - possible to make a single change and propagate automatically
  - represents the ultimate in content/representation separation

# Modularity & Style Sheets

```
<html>
<!-- CS443 page26.html 17.10.14 -->

<head>
  <title>Title for Page</title>
  <link rel="stylesheet"
        type="text/css"
        href="myStyle.css"
        title="myStyle" />
</head>

<body>
  <h1>Centered Title</h1>

  <p class="indented">This paragraph
will have the first line indented, but
subsequent lines will be flush.</p>

  <p>This paragraph will not be
indented.
</p>

  <h1>The End</h1>

</body>
</html>
```

view page

```
/* myStyle.css   CS443 02.09.05 */

h1 {color : blue; text-align :
center}
p.indented {text-indent:0.2in}
```

- ❖ Ideally, the developer(s) of a Web site would place all formatting options in an external style sheet.
- ❖ All Web pages link to that same style sheet for a uniform look.
- simplifies Web pages since only need to specify structure/content tags
- Note: no <style> tags are used in the external style sheet

# <div> and <span> Tags

- Problem: font properties apply to whole elements, which are often too large
  - Solution: a new tag to define an element in the content of a larger element - `<span>`
  - The default meaning of `<span>` is to leave the content as it is (i.e. unchanged)

```
<p> Now is the <span> best time </span> ever! </p>
```

- Use `<span>` to apply a document style sheet definition to its content

```
<style type = "text/css">  
  .bigred {font-size: 24pt;  
  font-family: Ariel; color: red}  
</style>  
... ..  
<p> Now is the <span class="bigred">  
  best time </span> ever!  
</p>
```

- The `<span>` tag is similar to other HTML tags, they can be nested and they have id and class attributes

[view page](#)

- Another tag that is useful for style specifications: `<div>`

Used to create document sections (or divisions) for which style can be specified e.g., a section of five paragraphs for which you want some particular style

# Web rules of thumb (ok, my rules of thumb...)

- HTML and CSS provide lots of neat features,  
but just because you can add a feature doesn't mean you should!  
don't add features that distract from the content of the page
  - use color & fonts sparingly and be careful how elements fit together  
e.g., no purple text on a pink background, no weird fonts  
e.g. I find bright white text on a black background difficult to read  
Consider the needs of visually impaired users of your website!!
  - use images only where appropriate  
e.g., bright background images can make text hard to read  
e.g., the use of clickable images instead of standard HTML buttons or links can slow access
  - don't rely on window or font size for layout  
e.g., font size may be adjusted by viewer, window constrained
  - don't be annoying  
e.g., lots of pop-up windows, excessive advertising, silly music
  - break a large document into several smaller ones or provide a menu for navigation
  - stick to standard features and test as many browsers as possible (and versions of the same browser)
  - utilize style sheets to make changes easy & ensure consistency

# Outline

## 1. Basic HTML

## 2. **Advanced HTML**

- HTML5 tags and attributes
- forms & inputs

# HTML5 Tags

- ❖ HTML5 has lots of flexibility:
  - Uppercase tag names
  - Quotes are optional for attributes
  - Attribute values are optional
  - Closing empty elements are optional
- ❖ HTML5 tag names are case insensitive
  - convention is to use all lower case
- ❖ HTML added support for multimedia

# HTML5 Attributes

- ❖ Elements may contain attributes
  - used to set various properties of an element.
- ❖ Some attributes are defined globally and can be used on any element, others are defined for specific elements only.
- ❖ All attributes have a name and a value
- ❖ Attributes defined in CSS
- ❖ Attributes may only be specified within start tags and must never be used in end tags.
  - attributes are case insensitive
  - convention is to stick with lower case.

# HTML5 New Tags

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>...</title>
  </head>
  <body>
    <header>...</header>
    <nav>...</nav>
    <article>
      <section>...</section>
    </article>
    <aside>...</aside>
    <figure>...</figure>
    <footer>...</footer>
  </body>
</html>
```

❖ HTML 5 specify DOCTYPE as follows:

**<!DOCTYPE html>**

❖ specify Character Encoding as follows:

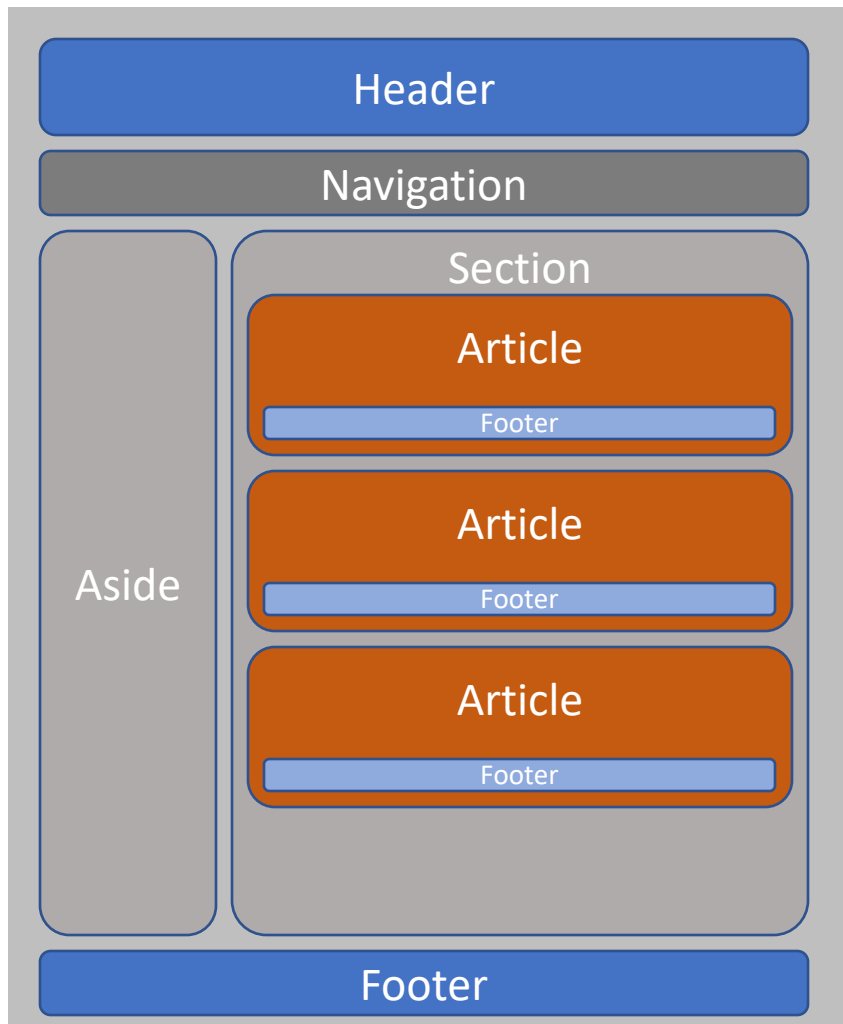
**<meta charset="UTF-8">**

❖ New tags introduced in HTML5 for better structure

- **header** – This tag represents the header of a section.
- **footer** – This tag represents a footer for a section and can contain information about the author, copyright information, etc.
- **nav** – This tag represents a section of the document intended for navigation.
- **dialog** – This tag can be used to mark up a conversation.
- **figure** – This tag can be used to associate a caption together with some embedded content, such as a graphic or video.



# HTML5 New Tags



- **section** – This tag represents a generic document or application section. It can be used together with h1-h6 to indicate the document structure.
- **article** – This tag represents an independent piece of content of a document, such as a blog entry or newspaper article.
- **aside** – This tag represents a piece of content that is only slightly related to the rest of the page.

# HTML5 New Tags

❖ HTML5 offers new elements for media content:

Tag	Description
<audio>	Defines sound content
<video>	Defines a video or movie
<source>	Defines multiple media resources for <video> and <audio>
<embed>	Defines a container for an external application or interactive content (a plug-in)
<track>	Defines text tracks for <video> and <audio>

```
<audio controls="true">  
  <source src="audiodemo.ogg" />  
  <source src=" audiodemo.mp3" />  
  <source src=" audiodemo.wav" />  
  Not supported.  
</audio>
```



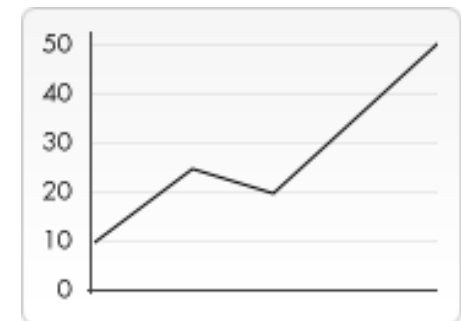
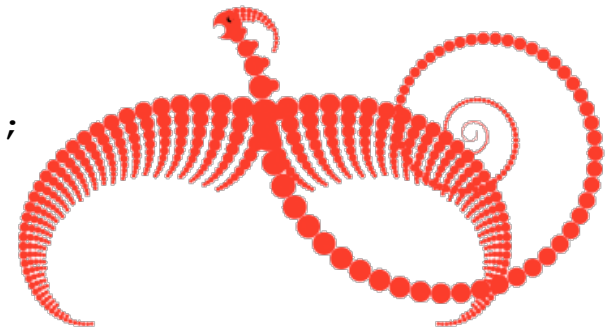
```
<video src="video.ogv" controls poster="poster.jpg"  
width="320" height="240">  
  <a href="video.ogv">Download movie</a>  
</video>
```

# HTML5 New Tags

## ❖ <canvas> element:

Tag	Description
<canvas>	Used to draw graphics, on the fly, via scripting (usually JavaScript)

```
function draw() {  
    var ctx =  
    document.getElementById('canvas').getContext('2d');  
    var img = new Image();  
    img.onload = function(){  
        ctx.drawImage(img,0,0);  
        ctx.beginPath();  
        ctx.moveTo(30,96);  
        ctx.lineTo(70,66);  
        ctx.lineTo(103,76);  
        ctx.lineTo(170,15);  
        ctx.stroke();  
    }  
    img.src = 'images/backdrop.png';  
}
```



# HTML5 New Tags

## ❖ New input elements:

button

checkbox

color

date

datetime

datetime-local

email

file

hidden

image

month

number

password

radio

range

reset

search

submit

tel

text

time

url

week

### • Form

Number:

Range:

Month:

DateTime:

UTC

DateTime-Local:

Date:

Submit

Week	Mon	Tue	Wed	Thu	Fri	Sat	Sun
17	26	27	28	29	30	1	2
18	3	4	5	6	7	8	9
19	10	11	12	13	14	15	16
20	17	18	19	20	21	22	23
21	24	25	26	27	28	29	30
22	31	1	2	3	4	5	6

Today None



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG  
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

**Thank you  
for your  
attentions!**

