Internet Programming- **HTML**

November 2013

HyperText Markup Language (HTML)

- A markup language, i.e. it's used to markup content.
- Used to tell the browser what to do, and what props to use and having series of tags that are integrated into a text document
- It is the lingua franca for publishing hypertext on the World Wide Web
- Allow to embed other scripting languages to manipulate design layout, text and graphics
- Not case sensitivity.
- Current version is **HTML5**

Creating and Viewing an HTML document?

1. Use an text editor such as Notepad, to write the document
2. Save the file as filename.html on a PC. This is called the HTML Document Source
3. Open internet Explorer (or any browser) Off-Line
4. Switch to internet Explorer
5. Click on File, Open File and select the filename.html document that you just created
6. Your HTML page should now appear just like any other Web page in internet Explorer

HTML Document

- Composed of several tags: the first tag turns the action on, and the second turns it off and the second tag(off switch) starts with a forward slash

- Should have an **.htm** or **.html** file name extension

- Can be created using
  1. Text Editors
     - Notepad, WordPad
  2. WYSIWYG Editors
     - Micro Soft FrontPage
     - Macromedia Dreamweaver

HTML Document (cont'd)

- HTML pages are tag-based documents
  - Really plain ASCII text files
  - Don't look like documents they represent
  - Tags indicate how processing program should display text and graphics
  - Designed to describe hypertext, not paper
  - Processed by browsers "on the fly"
  - Tags usually appear in pairs
  - Most have reasonable names or mnemonics
  - Most can be modified by attributes/values

HTML Document Structure

- **The HTML document is divided into two major parts:**
  - **HEAD**: contains information about the document:
    - Title of the page (which appears at the top of the browser window)
    - Meta tags: used to describe the content (used by Search engines)
    - JavaScript and Style sheets generally require statements in the document Head
  - BODY: Contains the actual content of the document
    - This is the part that will be displayed in the browser window

HTML Document Structure (cont'd)

- **General Structure of HTML files:**

```
❖<html>                                    →Start Tag
    <head>
        <title> Web page title
     </title>
    </head>
    <body>
  statement
        .......
    </body>
❖</html>                                   → End Tag
```

HTML Document Structure (cont’d)

- **Example1 HTML code:**

```
<HTML>
<head>
<title>Hello World</title>
</head>
<body bgcolor = “#000000”>
<font color = “#ffffff”>
<H1>Hello World</H1>
</font>
</body>
</HTML>
```

HTML Elements

- Names enclosed in < and >
- Commonly have a start tag and end tag
  - Start tag format: <tag_name>
  - End tag format: </tag_name> [note the / after <]
  - E.g. <strong>bold text</strong>
- Some tags may not have end tags
  - E.g. <br>
- Tags may have attributes
  - <tag_name attr1="val1" attr2="val2" …>…</tag_name>
  - E.g. <font face="arial" size="9">Hello</font>
- Not case sensitive

Type of tags

- **Metadata tags-** <title> ,<base>, <link>,<meta>. <style>
- **Section tags**: <body>, <head> , <div> ,<frameset> ,<frame>, <h1>..<h6> ,<p>
- **Text-level appearance tags:** <b>,<em> ,<strong>, <del>,<sub>
- **Grouping tags:** <dl>, <dt>, <ol>, <li>, <select>, <option>
- **Image tag:** <img>
- **Anchor tag:** <a>
- **Table tag**: <table> ,<th> ,<tr> ,<td>
- **Script tag:** <script>
- **Embeded content tags**: <applet>,<object>
- **Other tags:** <br> ,<hr>

Basic HTML Tags

- **Html-** everything in the document should be within <html> &</html>
- **Head-** contains information which is not displayed in the browser display area such as action-scripting (Javascript), styles (CSS) and general information referenced in the whole document
  - may contain other tags in it
  - format: <head>…</head>
- **Title-** sets the title of the web page to be displayed in the browser's title bar.
  - found within the <head> tag.
  - format: <title>…</title>

HTML Tags (cont'd)

- **Body:**
  - contains the visible part of the web page
  - displayed in the display area of the browser
  - contains several other tags and content in it
  - format: <body>…</body>
  - attributes:
    - bgcolor="color"
    - background="img url"
    - text="text color"
    - link="link color"
    - alink="active link color"
    - vlink="visited link color"
    - …

HTML Tags (cont’d)

- **Headings:**
  - predefined formats for text presentation
  - six heading formats defined in HTML: <h1> up to <h6>
    - <h1> the largest font size
    - <h6> the smallest font size
  - Format:
    - <h1>…</h1>
  - E.g. <h2>a text in heading two</h2>
  - bold
    - makes a text appear in bold
    - Format: <b>…</b> or <strong>…</strong>
    - E.g. <b>a text in bold</b>

HTML Tags (cont'd)

- italics
  - makes a text appear in italics
  - Format: <i>…</i>   or   <em>…</em>
  - E.g. <i>a text in italics</i>
- underline
  - makes a text appear underlined
  - Format: <u>…</u>
  - E.g. <u>underlined text</u>

- **Paragraph:**
  - defines a paragraph
  - Format: <p>…</p>

HTML Tags (cont’d)

- E.g. <p>this is a paragraph of text. it has a new line before and after it.</p>
- The browser inserts a new line before and after the text in the paragraph tag.
- attribute:
  - align=“alignment” {left, right, center, justify}

- **line break:**
  - inserts a new line
  - Format: <br>
  - E.g. line one <br> line two <br> line three <br> line four

HTML Tags (cont'd)

- horizontal rule
  - inserts a horizontal line
  - Format: <hr>
  - attributes:
    - width="width" {absolute: in pixels or relative: in %}
    - noshade
    - color="color" {browser dependent}
  - E.g. <hr width="75%" noshade color="#FF0000">
- sub/sup
  - define either a subscript or a superscript
  - Format: <sub>…</sub> ; <sup>…</sup>
  - E.g. X<sub>1</sub><sup>2</sup> + 2X<sub>3</sub>

HTML Tags (cont'd)

- **Lists**
  - Unordered Lists (ul)
    - define bulleted lists
    - Format:

```
<ul>
        <li>…</li>
        <li>…</li>
        …
</ul>
```

    - Atribute:
      - type="bullet type" {disc, circle, square}
    - E.g.

      <ul type="square"> <li>book</li><li>marker</li><li>chalk</li></ul>

HTML Tags (cont'd)

- Ordered Lists (ol)
  - define numbered lists
  - Format:

```
<ol>
        <li>…</li>
        <li>…</li>
        …
</ol>
```

  - Atribute:
    - type="number type" {1, i, I, a, A}
  - E.g.

  <ol type="i"> <li>book</li><li>marker</li><li>chalk</li></ol>

HTML Tags (cont'd)

- Definition Lists (dl)
  - define a list of term-description pairs
  - Format:

```
<dl>
    <dt>…</dt>
    <dd>…</dd>
    <dt>…</dt>
    <dd>…</dd>
    …
</dl>
```

  - E.g.

```
<dl>
    <dt>book</dt><dd>something that we read …</dd>
    <dt>marker</dt><dd>something we write with …</dd>
</dl>
```

HTML Tags (cont'd)

- **Images**
  - insert images in an html document
  - Format: <img> {no end tag}
  - Attributes:
    - src="img url"
    - alt="alternate text"
    - border="border width"
    - width="image width"
    - height="image height"
  - supported image formats:
    - gif, jpg/jpeg, png
  - E.g. <img src="assets/images/logo.gif" alt="Site Logo">

HTML Tags (cont'd)

- **Anchor**
  - defines a hyperlink or a named anchor
  - used for navigation
  - Format: <a>…</a>
  - Attributes:
    - href="url"
    - target="target" { _self, _blank }
    - name="anchor name"
  - E.g.

<a href="home.htm">Go to home</a>

<a href="http://www.google.com" target="_blank">Google</a>

HTML Tags (cont’d)

- Navigation with anchors
  - named anchors
    - named places in an html document
    - Format: <a name=“anchor_name”></a>
    - E.g. <a name=“top”></a>
  - linking to anchors
    - Format:
      - <a href=“#anchor_name”>link text</a> {on the same page}
      - <a href=“url#anchor_name”link text</a> {on a different page}
    - E.g.
      - <a href=“#top”>Top of page</a> {assuming the example above}
      - <a href=“http://www.hu.edu.et/history.htm#establishment”> Establishment of HU</a>

HTML Tags (cont'd)

- **Tables**
  - Insert tabular data
  - Design page layout
  - Powerful, flexible information delivery
  - Used to reflect or impart structure
  - A table is a container

    ```
    <table> ... </table>
    ```
  - Tags involved: <table>, <tr>, <td>, <th>, <caption>

HTML Tags (cont'd)

- Format:

```
<table>
    <caption>table caption</caption>
    <tr>
            <td>…</td> <td>…</td> …
    </tr>
    <tr>
            <td>…</td> <td>…</td> …
    </tr>
    …
</table>
```

HTML Tags (cont'd)

- E.g.

```
<table>
        <caption align="center" valign="bottom">table 1.0</caption>
        <tr>
                <th>Column 1</th> <th>Column 2</th>
        </tr>
        <tr>
                <td>Cell 1</td> <td>Cell2</td>
        </tr>
        <tr>
                <td>Cell 3</td> <td>Cell 4</td>
        </tr>
    </table>
```

HTML Tags (cont’d)

- Table attributes:
  - align=“alignment” {left, center, right}
  - bgcolor=“color”
  - width=“table width” {absolute or relative}
  - border=“border width”
  - bordercolor=“color”
  - cellspacing=“space amount” {in pixels}
  - cellpadding=“padding amount” {in pixels}
  - …

HTML Tags (cont'd)

- Table row attributes:
  - align="alignment" {left, center, right}
  - bgcolor="color"
  - height="height"
  - valign="alignment" {top, middle, bottom}
- Table data/heading attributes:
  - align="alignment"
  - valign="alignment"
  - width="width"
  - bgcolor="color"
- Unless otherwise specified, <tr> and <td> inherit attributes of <table> whenever it applies.

HTML Tags (cont'd)

- Cells spanning multiple rows/columns
  - two additional attributes of <td> and <th>
    - colspan="num columns"
    - rowspan="num rows"
  - E.g. (colspan)

```
<table>
    <tr> <td colspan="2">Cell 1</td> <td>Cell 2</td> <td>Cell 3</td>  </tr>
    <tr> <td>Cell 4</td> <td>Cell 5</td> <td>Cell 6</td> <td>Cell 7</td> </tr>
    <tr> <td colspan="4">Cell 8</td> </tr>
</table>
```

HTML Tags (cont'd)

- E.g. (rowspan)

```
<table>
    <tr> <td rowspan="3">Cell 1</td> <td>Cell 2</td> <td>Cell 3</td>  </tr>
    <tr> <td>Cell 4</td> <td>Cell 5</td> </tr>
    <tr> <td>Cell 6</td> <td>Cell 7</td> </tr>
</table>
```

- E.g. (hybrid)

```
<table>
    <tr> <th colspan="3">Title</th> </tr>
    <tr> <th rowspan="3">Cell 1</th> <td>Cell 2</td> <td>Cell 3</td> </tr>
    <tr> <td>Cell 4</td> <td>Cell 5</td> </tr>
    <tr> <td>Cell 6</td> <td>Cell 7</td> </tr>
</table>
```

HTML Tags (cont’d)

- HTML comments
  - insert commented text in an html document
  - Format: <!-- comment text -->
  - E.g. <!-- this is a comment text -->

HTML Special Characters

- Special characters (named characters)
  - not found on the standard keyboard
    - e.g. ©
  - used by HTML
    - e.g. <
  - ignored by browsers
    - e.g. blank spaces
- Format:
  - **&code;**
- Examples:
  - &copy; ➔ © &lt; ➔ < &amp; ➔ & &nbsp; ➔ space

HTML Forms

Divisions

- In HTML, we can create divisions of an HTML document using the **<div>** tag.
- A <div> is a logical block tag that has no predefined meaning or rendering
- Very important for page layout design
- The <div> tag works well with CSS
- Tag format:
  - <div> … </div>
- Attributes:
  - align="alignment" {left, right, center} - define content alignt.

HTML Forms

- Used to gather input from users
- The input is usually sent to a **server-side script** for processing

- The data can be sent in two methods: GET & POST
- GET
  - for small and non-secure data
  - Is the default method
  - Data is sent as part of the request URL
    - → Limitation in size of data

HTML Forms (cont’d)

- POST
  - For large & secure data
  - Input is sent as a data stream after the request URL
- Tags
  - The <form> tag
    - Contains all input elements in the form
    - Defines the method of sending data
    - Defines the server-side script responsible for accepting the data

HTML Forms (cont'd)

- Attributes:
  - name="name"
  - method="method" {get, post}
  - action="url" {url of the server-side script to post data to}
  - enctype="enctype" {multipart/form-data, text/plain, … }
    - multipart/form-data – used when uploading files

- Ex.

```
<form method="post" action="search.php">
….
</form>
```

HTML Forms (cont’d)

- <input> tag
  - used to define input fields in a form
  - several types of input fields
    {textbox, listbox, checkbox, radio, button, …}
- Attributes:
  - name=“name”
  - type=“type”
    {text, hidden, password, file, submit, reset, button, checkbox, radio, image}
  - value=“value”
  - disabled=“disabled”
  - checked=“checked”

HTML Forms (cont'd)

- The possible input types:
  - **text** – input text box
  - **hidden** – a hidden field for storing values
  - **password** – password input box
  - **file** – input box for file uploading (browse)
  - **submit** – a button that submits the form
  - **reset** – a button that resets form fields to their original values
  - **button** – a general purpose button
  - **checkbox** – a check box
  - **radio** – a radio button (option button)
  - **image** – an image button that submits the form

HTML Forms (cont’d)

- Other input fields
  - <textarea> tag
    - used to input a large block of text
  - Tag format: <textarea>…</textarea>

  - Attributes:
    - name=“name”
    - cols=“num_columns”
    - rows=“num_rows”
    - readonly=“readonly”
    - wrap=“wrap_type” {off, hard, soft, virtual, physical}

HTML Forms (cont'd)

- <select> tag
  - used to create a select box
- Tag format:
  - <select>

```
      <option>…</option>
       <option>…</option>
      …
  </select>
```

- Attributes:
  - <select>
    - name="name"
    - multiple="multiple" {enables selection of multiple items}
    - disabled="disabled"

HTML Forms (cont'd)

- <option>
  - value="value"
  - selected="selected"
  - disabled="disabled" {browser compatibility: Firefox ?}

- Ex.

1. <select name="department">

   <option value="1">Computer Science</option>

   <option value="2">Information Science</option>

   <option value="3">Computer Engineering</option>

   </select>
2. Modify the above input so that **Information Science** is selected by default.

HTML Forms (cont'd)

- submit & reset buttons
  - the vlaue attribute is the caption of the buttons

  Ex. <input type="submit" value="ok">

  →inserts a button with the caption (label) **ok**.

- file upload fields

  Ex. <input type="file" name="doc">

HTML Forms (cont’d)

- <label> tag
  - used to give labels to input fields

  - Ex.

```
<label>First Name:
    <input type=“text” name=“fname”>
</label>
```

HTML Forms (cont'd)

- <fieldset> tag
  - used to group input fields in a form
  - the title/label of the group is set using the <legend> tag
  - Tag format:

```
<fieldset>
   <legend>…</legend>
   … one or more input fields …
</fieldset>
```

  - Attributes:
    - <legend>
      - align="alignment" {left, center, right}

HTML Forms (cont'd)

- Presentation
  - tables can be used to align form fields and their labels so that the presentation of the form looks more regular.
  - one possibility is to use:
    - one table row per input field
    - within a table row
      - □ one column for the labels
      - □ one column for the input fields
  - the table borders can be set to 0 so that they are not visible
  - other features of tables (rows & columns) can be adjusted as required

HTML Forms (cont’d)

- Presentation (cont’d)
  - Cascading Style Sheets (CSS) can be used to define further presentation related attributes for input fields
  - Ex. all text input fields should have font size of 10 and background color of grey.
  - will be discussed in the next section.

HTML Forms (cont'd)

- Exercises:

1. Create an HTML page which displays a login screen with
   - ☐ a **username** field
   - ☐ a **password** field
   - ☐ a button to submit the form
   - ☐ a button to reset the content of the form

HTML Forms (cont'd)

2. Create an HTML page which displays student registration screen with
   - a **name** field
   - an **ID** field
   - a **department** field {select box with values:
     1. Computer Science
     2. Information Science
     3. Computer Engineering
   - a **semester** field {select box with values:
     1. I
     2. II
   - an **academic year** field
     {select box with values: 1998-2000, 2000 default}
   - a button to submit the form
   - a button to reset the form

Frames

Frames & Framesets

- Frames are a way of dividing the browser window into several independent windows in which can be loaded different urls.
- Each frame can be independently loaded a different url.
- Each frame can scroll independently when necessary
- Has advantages and drawbacks
- Advantages
  - Improved performance (minimal page refresh)
  - Flexibility
  - Simultaneous multiple views

Frames & Framesets (cont'd)

- Drawbacks
  - Fairly complex (for developer)
  - May confuse users (if not properly used)
  - Some (old) browsers may not support frames
  - URL masking, when printing and bookmarking

- Frames are found in Framesets
- Framesets define the 'layout' of the frames it contains
- Several frames can be included in a frameset
- Framesets can be nested in one another to provide a more complex layout.

Frames & Framesets (cont'd)

- Tag format:

  Single frameset

```
<frameset>
        <frame>
        <frame>
        …
</frameset>
```

Frames & Framesets (cont'd)

Nested framesets

```
<frameset>
        <frame>
        <frameset>
                <frame>
                <frame>
                …
        </frameset>
        <frame>
        …
  </frameset>
```

Attributes

- Frameset
  - rows = “row dimensions” { ex. rows=“10%, 90%” }
  - cols = “column dimensions” { ex. cols=“20%, *, 30% }
  - border = “value”
  - bordercolor = “color”
  - frameborder = “value” { 0, 1, no, yes }
  - framespacing = “value”

Attributes (cont’d)

- Frame
  - name = “frame_name”
  - src = “url” { url can be local or external }
  - noresize [= “noresize”]
  - scrolling = “value” { auto, yes, no }
  - allowtransparency = “value” { in % }
  - frameborder = “value” { 0, 1, yes, no }
  - bordercolor = “color”

Examples

index.html

```
<html><head><title>My site</title></head>
<frameset cols = “30%, 70%”>
    <frame src=“nav.html”  noresize scrolling=“no”>
    <frame name = “viewer” src=“http://www.google.com”  noresize>
</frameset>
</html>
```

nav.html

```
<html><body>
<a href=“http://www.google.com” target=“viewer”>google</a><br>
<a href=“http://www.yahoo.com” target=“viewer”>yahoo</a><br>
<a href=“http://www.altavista.com” target=“viewer”>altavista</a><br>
<a href=“http://www.msn.com” target=“viewer”>msn</a><br>
</body></html>
```

preview

Examples (cont'd)

index.html

```
 <html><head><title>My site</title></head>
<frameset cols = “30%, 70%”>
    <frame src=“nav.html” scrolling=“no”>
    <frameset rows = “20%,  *”>
        <frame src=“title.html”  noresize>
        <frame name = “viewer” src=“http://www.google.com”  noresize>
    </frameset>
</frameset>
</html>
```

title.html

```
<html><body>
        <h1>Welcome to My Bookmarks !</h1>
</body></html>
```

preview

Inline Frames (iframe)

- iframes are also called floating frames
- Like frames
- But can occur anywhere in the <body> tag of an html document
  - Unlike frames which should only occur in the <frameset> tag

- Tag format:

  <iframe></iframe>

iframe (cont’d)

- Attributes
  - src = “url”
  - name = “name”
  - height = “value”
  - width = “value”
  - scrolling = “value” { auto, yes, no }
  - align = “alignment” { left, right, middle, top, bottom }
  - allowtransparency = “value”

Example

```
<html>
   <head>
         <title>iFrame sample</title>
   </head>
   <body>
         Below is the google site <br>
         <iframe height="600px" width="600px"
         src="http://www.google.com"
         scrolling="auto"></iframe>
   </body>
</html>
```

HTML5

HTML5 Overview

- HTML5 is the latest and the most enhanced version of html that aims to make HTML more useful for creating internet and web applications as well as semantically marked up documents
- It is the standard for structuring and presenting content on the WWW
- HTML5 offers new features (elements, attributes, event handlers, and APIs) for easier web application development and more sophisticated form handling.
- New Features such as :Video and Audio playback, Drag- and –Drop, Canvas, Server-sent Events ,etc are included.

HTML5 Overview (cont'd)

- HTML5 is designed to deliver almost everything you want to do online without requiring additional plug-ins.
  - It does everything from animation to apps, music to movies, and can also be used to build complicated applications that run in your browser.
- HTML5 is also cross-platform and become part of the future of the Web.
- It is used to write web applications that still work when you are not online.

HTML5-Rules

- New features should be based on HTML, CSS, DOM, and JavaScript
- The need for external plug-ins (like Flash) needs to be reduced
- Error handling should be easier than in previous versions
- Scripting has to be replaced by more markup
- HTML5 should be device-independent
- The development process should be visible to the public

New in HTML5

- HTML5 introduces the following **new input and form control** types : calendar, date, time, email, url, search
- **APIs:** With a growing demand for interactive content on web pages,HTML5 introduces several APIs (Application Programming Interfaces) for standardizing the creation of web applications.
  - Drag and drop functionality (including the new draggable attribute)
  - Playing video and audio files, with video and audio elment
  - Two-dimensional drawing in conjunction with the new **canvas** element
  - Registering applications for certain protocols or media types
  - Cross-document messaging
  - New content-specific elements, like <article>, <footer>, <header>, <nav>, <section>

New HTML5- Global attributes

- In addition to id, class, style, title, dir, lang, accesskey, and tabindex carried over from HTML 4.01, HTML5 adds the following global attributes that are applicable to all elements:
  - contenteditable="true|false“
  - contextmenu="id of menu element“
  - draggable="true|false“
  - spellcheck="true|false"

Basic structure of an HTML5 document

```
<!DOCTYPE html>
<html>
  <head>
<meta charset="UTF-8">
    <title>Document Title</title>
  </head>
  <body>
      Content of document . . .
  </body>
</html>
```

Exercises

1. Create an HTML document with the following elements:
   - At least one image
   - At least three links
   - Different background color and text colors.
   - Two different sizes of text
   - Some centered text.
   - A title

Exercises (cont'd)

2. For this exercise, you will familiarize yourself with HTML by hand-writing a web site. You will develop at least two web pages.
   - The first will provide a Google-like interface where a user can perform a search query.
   - The "Search" button will link to the second page that displays the results of a query.
   - You will not actually be generating results. Instead, your "Search" button can simply link to another page that shows the results for a static query.
   - The goal is really just to design the interface.

Exercises (cont'd)

3. Create a menu list of links to the following search engines:
   - Yahoo: http://www.yahoo.com
   - Google: http://www.google.com
   - Alta Vista: http://www.altavista.com
   - Ask: http://www.ask.com

- Refer to the following syntax:

```
<a href=”<URL>”>Text to Display</a>
```

Exercises (cont'd)

- Differentiate HTML vs. XML in details

- Next CSS