Content

- Hypertexts
- Architectural overview of the Web
- Web browser
- Thin vs. Thick clients
- Servers
- HTML: Hypertext Markup Language
- URL: Uniform Resource Locator
- Basic HTML tags and attributes
- The meta tag
- HTML validation
- Content vs. Presentation
- Styles: CSS Cascading Style Sheets
Hypertext

- **Hypertext** is text which is not constrained to be linear.
- Hypertext is text which contains links to other texts.
- **Link**: a relationship between two anchors, stored in the same or different text.
- **Anchor**: an area within the content of a node which is the source or destination of a link - the anchor may be the whole of the node content.
- **Node**: a unit of information.
- The term was coined by Ted Nelson around 1965.
- **HyperMedia** is a term used for hypertext which is not constrained to be text: it can include graphics, video and sound, for example.

http://www.w3.org/2003/glossary/

http://www.w3.org/Terms.html
Architectural Overview

The parts of the Web model
When you click on a http://www.unibz.it

- The browser determines the URL (sees what is selected)
- The browser asks DNS for the IP address of www.unibz.it
- DNS replies with 193.206.186.140
- The browser makes a TCP connection to port 80 on 193.206.186.140
- It sends over a request asking for path "/" and default filename
- The www.unibz.it server sends the file /index.html
- The TCP connection is released
- The browser displays all the text in index.html (formatting the text according to the instructions contained in the page).
Thin vs. Thick Clients

- **Web browser**: software that allows the user to view certain types of Internet files in an interactive environment
  - Internet Explorer
  - Firefox
  - Opera
  - Safari
- Web Apps are (typically) “Thin”
- **Server does processing**
- **Client does presentation**
  + Simple! (Browser)
  - Limited GUI (HTML).
Thin vs. Thick Clients

- Software is “Thick”
  - E.g., a word processor
- Thick **clients** do **processing** and **presentation**
  - + GUI not limited by HTML
  - + Snappy (fewer Latency Problems)
  - — People need to download & install client
- Example (thick) client: Java Applets
  - Java applications running on the Java virtual machine included in the browser
  - You must "download" the java plugin to run Java applets.
**Mappa Mibtel**

[La mappa dell'indice di tutti i titoli della borsa italiana]

Ultimo aggiornamento: 13:18 - 13/02/2009
Thick Email Client

Start of email:

Request for Research Opportunity
richa@itgernet.in

Dear Prof. Francesco Pizzol,

I write to introduce myself as a 3rd year student of Indian Institute of Technology (IIT) Guwahati, India majoring in Communication Design. As a part of the curriculum, I am required to go on an internship for a period of 10 - 12 weeks during summer 2008 beginning second week of May till July.

I am interested in working in the field of "Human Computer Interaction, Usability Engineering, Interface Design, Web and Mobile Based Application". I have been pursuing research after my B-Tech and the projects undertaken by me would be very useful for my final year project and higher studies.

I have also done my MDP in Design (Interactive) at IIT. My project is called "Intelligent Wall Hanging Lamps". The idea for building this interactive product and Intelligent Environment derived from the observation of the non-occupational activities and how humans interact with their environment.

During my semester project of 5th semester I have done a project on "Determining the influence of mobile phone features for different age group of Indian Women" got selected for the Symposium sponsored by Danish Council of Independent Research (DFF) & ITF.

I am looking for an interesting opportunity to work on your project which I will find highly interesting.

I have attached the final report of my MDP Project for your reference.

I look forward to hearing your early response.

Sincerely,

Richa
Thin Email Client

Request for Research Opportunity
richa@iltg.ernet.in [richa@iltg.ernet.in]

Follow up
Gli allegati possono contenere virus dannosi per il computer. Gli allegati potrebbero non essere visualizzati in modo corretto.

A: Rico Francesco (F)

C:

Allegati:

[Embed image]

Dear Prof. Francesco Rico,

I introduce myself as a 3rd year student of Indian Institute of Technology (IIT) Guwahati, India majoring in Communication Design. As a part of the curriculum, I am required to go on an Internship for a period of 10 - 12 weeks during summer 2002 beginning second week of May till July.

I am interested in working in the field of "Human Computer Interaction, Usability Engineering, Interface Design, Web and Mobile Based Applications." I am keen on pursuing research offer by B.Tech and the projects undertaken by me would be very useful for my final year project and higher studies.

I came to know about you through the website <http://www.iltg.ernet.in> - I was particularly interested by your projects and publication. I find your project "Packaging" highly interesting.

During my semester project of 3rd semester I have done a project on "Intelligent Wall Hanging Lamp". The idea for building this interactive product and Intelligent Environment derived to bring computation into the real physical world to support what is traditionally considered non-computational activities and to allow human interaction.
The Client Side

- (a) A browser plug-in
- (b) A helper application

The browser decides what to do based on the **Internet media type** (previously called MIME) of the response: e.g., image/gif (see details in a next lecture)
Lectures

- **Lecture 1 - Introduction to the seminars**: description of the course objectives and exam procedure. Introduction to the syllabus topics and short presentation of the papers illustrating the problem considered and the approach used. [pdf] - Room E411 - Wed. October 3, 14:00 - 17:00
- **Lecture 2 - Linear Algebra and Markov Chains**: linear algebra, matrices, eigenvalues and eigenvectors, markov chains, Google PageRank. [pdf] - Room A101 - Wed. Oct. 10, 14:00 - 17:00
- **Lecture 3 - Information Retrieval**: information retrieval, Web search, indexing, document model, relevance, evaluation of an information retrieval system. [pdf] - Room D003 - Wed. Oct. 24, 14:00 - 17:00
- **Lecture 4 - Machine Learning for IR**: machine learning, support vector machines, recommender systems. [pdf] - Room A101 - Wed. Oct. 31, 14:00 - 17:00
  - More info on ML and support vector machines on: Tan, Steinbach & Kumar, Introduction to data mining, Addison Wesley, 2006.

Papers

   - Speakers: Diana Zverelo & Paulius Miksys. November 7
   - Speakers: Gregory Osunde & Tumas Cytsis. November 14
   - Speakers: Markus Innerebner & Paulius Miksys. November 21
Plug-in

Seminars in Databases

Course Overview and Motivation

- Acrobat pdf reader (plugin) has been invoked by the browser (the content-type of the response is application/pdf).
Now the helper will be invoked.
Changing the behavior of browser

- You can change how the browser will react to different content types (MIME).
Servers

- **Hardware server**
  - Computer on Internet, always running

- **Software server (aka daemon)**
  - Program running on server
  - Listening on port
    - Receives requests, processes them, makes outgoing calls
  - Daemon examples:
    - **sshd**: allow to exchange data over a secure channel (encryption)
    - **lpd**: line printer daemon (in Berkeley Unix)
    - **httpd**: the hypertext transfer protocol daemon (more on that after!)
What the server will do

- Basic model
  1. Accept a TCP connection from the client browser
  2. Get the name of the file requested
  3. Get the file from the disk
  4. Return the file to the client
  5. Release the TCP connection

- **Problem:** no more files/sec returned that file-access/sec (if the file is written in contiguous blocks)

- **Solution:** maintain a cache in memory of the most frequently accessed files.
## Hardware assumptions

<table>
<thead>
<tr>
<th>symbol</th>
<th>statistic</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>average seek time</td>
<td>5 ms = 5 x 10&lt;sup&gt;-3&lt;/sup&gt; s</td>
</tr>
<tr>
<td>b</td>
<td>transfer time per byte</td>
<td>0.02 µs = 2 x 10&lt;sup&gt;-8&lt;/sup&gt; s</td>
</tr>
<tr>
<td></td>
<td>processor’s clock rate</td>
<td>10&lt;sup&gt;9&lt;/sup&gt; s&lt;sup&gt;-1&lt;/sup&gt;</td>
</tr>
<tr>
<td>p</td>
<td>low-level operation (e.g., compare &amp; swap a word)</td>
<td>0.01 µs = 10&lt;sup&gt;-8&lt;/sup&gt; s</td>
</tr>
<tr>
<td></td>
<td>size of main memory</td>
<td>several GB</td>
</tr>
<tr>
<td></td>
<td>size of disk space</td>
<td>1 TB or more</td>
</tr>
</tbody>
</table>

**Example:** Reading a page of 100kB (10<sup>5</sup> B) from disk

- If stored in contiguous blocks: \(2 \times 10^{-8} \text{s} \times 10^5 + 5\text{ms} = 2\text{ms} + 5\text{ms} = 7\text{ms}\)
- If stored in 100 files: \(2\text{ms} + 100 \times 5 \times 10^{-3}\text{s} = 0.502\text{s}\)
The Server Side

- A multithreaded Web server with a front end and processing modules

- This is the model used by the Servlets (each servlet on a different thread).
1) Resolve the name of the Web page requested
2) Authenticate the client
3) Perform access control on the client
4) Perform access control on the Web page
5) Check the cache
6) Fetch the requested page from disk (if not in cache)
7) Determine the MIME type to include in the response (content-type header)
8) Return the reply to the client
9) Make an entry in the server log
A Web Farm

- Each time a request is made the front end dispatches it to one of the servers in the farm.
- Failure of individual machines is managed (redundancy and automatic failover).
Google Web Farm

- The best guess is that Google now has more than **450,000 servers** (2 Petabytes of RAM $2 \cdot 10^6$ Gigabytes)
- Spread over at least **25 locations** around the world
- Connecting these centers is a high-capacity fiber optic network that the company has assembled over the last few years.

Google is building two computing centers, top and left, each the size of a football field, in The Dalles, Ore.

J. Markoff, NYT, June 2006
## URLs – Uniform Resource Locators

### Some common URLs

<table>
<thead>
<tr>
<th>Name</th>
<th>Used for</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>http</td>
<td>Hypertext (HTML)</td>
<td><a href="http://www.cs.vu.nl/~ast/">http://www.cs.vu.nl/~ast/</a></td>
</tr>
<tr>
<td>file</td>
<td>Local file</td>
<td>file:///usr/suzanne/prog.c</td>
</tr>
<tr>
<td>news</td>
<td>Newsgroup</td>
<td>news:comp.os.minix</td>
</tr>
<tr>
<td>news</td>
<td>News article</td>
<td>news:<a href="mailto:AA0134223112@cs.utah.edu">AA0134223112@cs.utah.edu</a></td>
</tr>
<tr>
<td>gopher</td>
<td>Gopher</td>
<td>gopher://gopher.tc.umn.edu/11/Libraries</td>
</tr>
<tr>
<td>mailto</td>
<td>Sending e-mail</td>
<td><a href="mailto:JohnUser@acm.org">mailto:JohnUser@acm.org</a></td>
</tr>
<tr>
<td>telnet</td>
<td>Remote login</td>
<td>telnet://www.w3.org:80</td>
</tr>
</tbody>
</table>
Uniform Resource Locators URL

- Uniform Resource Locator (URL) is used to address a document (or other data) on the World Wide Web.
- A full Web address like this: http://www.w3schools.com/html/lastpage.htm follows these syntax rules:
  - scheme://host.domain:port/path/filename

  - The **scheme** is defining the **type** of Internet service: e.g. http or ftp or file.
  - The **domain** is defining the Internet **domain name** like w3schools.com.
  - The **host** is defining the domain host. If omitted, the default host for http is **www**.
  - The **:port** is defining the **port number** at the host. The port number is normally omitted. The default port number for http is **80**.
  - The **path** is defining a **path** (a sub directory) at the server.
  - The **filename** is defining the name of a document. The default filename might be default.asp, or index.html or something else depending on the settings of the Web server.

http://www.w3.org/Addressing/
A Uniform Resource Identifier (URI) provides a simple and extensible means for identifying a resource. A URI may be classified as:

- URN (Uniform Resource Name) is like a person's name,
- URL (Uniform Resource Locator) is like their street address

A Uniform Resource Locator (URL) is a URI that, in addition to identifying a resource, provides means of acting upon or obtaining it.

- Ex: the URL http://www.wikipedia.org/ is a URI that identifies a resource and implies that a representation of that resource (HTML code) is obtainable via HTTP from a network host named www.wikipedia.org.

A Uniform Resource Name (URN) is a URI that identifies a resource by name in a particular namespace.

- Ex: the URN urn:isbn:0-395-36341-1 is a URI that allows one to talk about a book, but doesn't suggest where and how to obtain an actual copy of it.
<html>
<head> <title> My New Web Page </title> </head>
<body>
<h1> Welcome to My Web Page! </h1>
<p> This page illustrates how you can write proper ... </p>
<p> There is a small graphic after the period at the end of this sentence. <img src="images/mouse.gif" alt="Mousie" width="32" height="32" border="0"> The graphic is in a file. The file is inside a folder named "images." </p>
<p> Link: <a href="http://www.yahoo.com/">Yahoo! </a></p>
<br> Another link: <a href="tableexample.htm">Another Web page</a>
<br> Note the way the BR tag works in the two lines above. </p>
<p>&gt; <a href="index.htm">HTML examples index</a></p>
</body>
</html>
HTML Versions

- **1992**
  - HTML is first defined

- **1993**
  - HTML+ (some physical layout, fill-out forms, tables, math)

- **1994**
  - HTML 2.0 (standard for core features)
  - HTML 3.0 (an extension of HTML+ submitted as a draft standard)

- **1995**
  - Netscape-specific non-standard HTML appears

- **1996**
  - Competing Netscape and Explorer versions of HTML
  - HTML 3.2 (standard based on current practices)

- **1997**
  - HTML 4.0 (separates structure and presentation with stylesheets)

- **1999**
  - HTML 4.01 (slight modifications only)

- **2000**
  - XHTML 1.0 (XML version of HTML 4.01)

- **2001**
  - XHTML 1.1 (modularization to allow different subsets)

- **2002**
  - XHTML 2.0 (simplifying and generalizing several tags)
HTML 5 (draft specification)

- HTML 5 introduces a number of new elements and attributes. Ex:
  - `<footer>` (usually referring to bottom of web page or to last lines of HTML code),
  - `<audio>` and `<video>` instead of `<object>`

- New APIs
  - **Offline storage database** (offline web applications)
  - **Geolocation**
  - **File API**, handle file uploads and file manipulation
  - **Directories and System**, to satisfy client-side storage
  - **File Writer**, for writing to files from web applications.
HTML Characteristics

- Just a Text File!
  - Portable
  - Human Readable/Writable

- **Defines the Structure** (not Appearance) of the document
  - **Client** (Browser) *defines the appearance*
    (based on the instructions in the html file and some other linked files)
  - Portable
  - Pours into Browser (PDAs, Bigger/Smaller)
Document Structure

<html>
<head>
<title>My First Web Page</title>
</head>
<body bgcolor="white">
<!-- This is a comment -->
<p>A Paragraph of Text.</p>
</body>
</html>

Specifications:  http://www.w3.org/TR/html401/
HTML Tags

- HTML tags are used to mark-up HTML elements.
- HTML tags are surrounded by the two characters `<` and `>` (angle brackets).
- HTML tags normally come in pairs like `<b>` and `</b>`.
- The first tag in a pair is the start tag, the second tag is the end tag.
- The text between the start and end tags is the element content.
- HTML tags are not case sensitive, `<b>` means the same as `<B>` (XHTML requires to use lowercase!).
Tag attributes

- Each element may have any number of **attributes**
  - **Ex:** `<body bgcolor="white">...</body>`
- Attributes provide **additional information** to an HTML element
- Attributes always come in **name/value pairs** like this: `name="value"`
- Attributes are always specified in the start tag of an HTML element
- Attributes and attribute values are also case-insensitive
- World Wide Web Consortium (W3C) recommends lowercase attributes/attribute values in their HTML 4 recommendation
- XHTML demands lowercase attributes/attribute values.
Nested Tags

- Like a tree, each element is contained inside a parent element

```
<html>...
  <head>...
    <title>...
  </head>
  <body>...
    <p>...
    <br>
    <table>...
  </body>
</html>
```

This is some text!
Basic Tags

- `<!doctype html public "-//w3c//dtd html 4.0 transitional//en">`
  - Preamble which identifies content as HTML
- `<h1>...</h1>`
  - H1 ... H6 : heading, where larger number means smaller heading
- `<p>`
  - Includes vertical whitespace (before and after the paragraph) unlike `<br>`
- `<hr>` horizontal rule
- `<br>` new line
- `<b>...</b>` bold
- `<i>...</i>` italicize text in between.
Lists

- **Unordered Lists**
  
  ```html
  <ul>
   <li> Apples
   <li> Oranges </li>
  </ul>
  ```

- **Ordered Lists**
  
  ```html
  <ol>
   <li> One
   <li> Two
  </ol>
  ```

- **Lists can be nested**
  
  ```html
  <ul>
   <li> Apples
    <ol>
     <li> Fuji
     <li> Granny Smith
    </ol>
   <li> Oranges
  </ul>
  ```
Comments and special characters

- <!-- This is a comment -->
- <!--
   This paragraph,
   is also a
   comment...
-->
- **Character Entities**
  - &lt; → <
  - &gt; → >
  - &amp; → &
  - &nbsp; → space
Anchor Tag (Links)

- **Absolute** HREFs specify fully qualified URLs
  - `<a href="http://www.yahoo.com/">Yahoo!</a>`
  - "a" means ANCHOR

- **Relative** HREFs are relative to the directory containing the current HTML file
  - `<a href="reldoc.html">In this directory!</a>`
  - `<a href="a/doc.html">In sub-directory a!</a>`

- With the **target** attribute, you can define **where** the linked document will be opened
  - Ex: open the link in a new window
    - `<a href="http://www.w3schools.com/" target="_blank">Visit W3Schools!</a>`
Tables

- Tables are defined with the `<table>` tag
- A table is divided into rows (with the `<tr>` tag)
- Each row is divided into data cells (with the `<td>` tag) - "table data"
- A data cell can contain text, images, lists, paragraphs, forms, horizontal rules, tables, etc.

```html
<table>...
<tr>... for each row
<td>... for each element in a row
<th>... for each element in the header row
</table>
```
Table Example

<table border="1">
<tr> <th>Heading</th> <th>Another Heading</th> </tr>
<tr> <td>row 1, cell 1</td> <td>row 1, cell 2</td> </tr>
<tr> <td>row 2, cell 1</td> <td>row 2, cell 2</td> </tr>
</table>

<table>
<thead>
<tr>
<th>Heading</th>
<th>Another Heading</th>
</tr>
</thead>
<tbody>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</tbody>
</table>
Layout

- One very common practice with HTML, is to use HTML tables to format the layout of an HTML page

```html
<table>
  <tr>
    <td>
      <b><font size="+3">Francesco Ricci</font></b>
      <p>
        <a href="http://www.unibz.it/">Free Univ. of Bozen-Bolzano</a>
        <br>
        <a href="http://www.unibz.it/inf/">Faculty of Computer Science</a>
      </p>
      <p>
        Piazza Domenicani 3, I-39100 Bozen-Bolzano, Italy<br>
        Phone: 0471 016 971, fax: +39 0471 016 009 <br>
        email: <a href="mailto:fricci@unibz.it">fricci At unibz.it</a>
      </p>
    </td>
    <td>
      <img src="images/FrancescoRicci.jpg" alt="A photo of Francesco Ricci">
    </td>
  </tr>
</table>
```
Colors

- HTML colors can be defined as a hexadecimal notation for the combination of Red, Green, and Blue color values (RGB).

<table>
<thead>
<tr>
<th>Color</th>
<th>Color HEX</th>
<th>Color RGB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black</td>
<td>#000000</td>
<td>rgb(0,0,0)</td>
</tr>
<tr>
<td>Red</td>
<td>#FF0000</td>
<td>rgb(255,0,0)</td>
</tr>
<tr>
<td>Green</td>
<td>#00FF00</td>
<td>rgb(0,255,0)</td>
</tr>
<tr>
<td>Blue</td>
<td>#0000FF</td>
<td>rgb(0,0,255)</td>
</tr>
<tr>
<td>Cyan</td>
<td>#FFFF00</td>
<td>rgb(255,255,0)</td>
</tr>
<tr>
<td>Yellow</td>
<td>#FF00FF</td>
<td>rgb(255,0,255)</td>
</tr>
<tr>
<td>Lime</td>
<td>#FFFF00</td>
<td>rgb(0,255,255)</td>
</tr>
<tr>
<td>Maroon</td>
<td>#C0C0C0</td>
<td>rgb(192,192,192)</td>
</tr>
<tr>
<td>White</td>
<td>#FFFFFF</td>
<td>rgb(255,255,255)</td>
</tr>
</tbody>
</table>

- W3C has listed 16 color names that will validate with an HTML validator: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, and yellow.

Image Files

- `<img src="URL of image file" alt="Big Boat" >`

- The URL points to the location where the image is stored (with usual conventions on URL)

- The alt attribute is used to define an "alternate text" for an image: the value of the alt attribute is an author-defined text

- If an HTML file contains ten images - eleven files (http requests) are required to display the page right!
Image Formats

- **JPEG**
  - Best for photos
  - Public standard

- **GIF**
  - Best for simple images
  - Older standard

- **PNG** – Portable Network Graphics
  - Public standard replacement for GIF

- **SVG** – Scalable **Vector** Graphics
  - Series of drawing commands
  - Uses XML
The head element contains general information, also called **meta-information**, about a document.

- The elements inside the head element (e.g., title, base, link, meta) should not be displayed by a browser.

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;head&gt;</td>
<td>Defines information about the document</td>
</tr>
<tr>
<td>&lt;title&gt;</td>
<td>Defines the document title</td>
</tr>
<tr>
<td>&lt;base&gt;</td>
<td>Defines a base URL for all the links on a page</td>
</tr>
<tr>
<td>&lt;link&gt;</td>
<td>Defines a resource reference (e.g. css)</td>
</tr>
<tr>
<td>&lt;meta&gt;</td>
<td>Defines meta information</td>
</tr>
</tbody>
</table>
HTML head - base

- Assume that the absolute address for an image is:
  `<img src="http://www.w3schools.com/images/smile.gif" />

- Now we insert the `<base>` tag, which specifies a base URL for all of the links in a page, in the head section of a page:
  `<head>
  <base href="http://www.w3schools.com/images/" />
  </head>

- When inserting images on the page in the example above, we just have to specify the **relative** address
  `<img src="smile.gif" />

- The browser will look for that file using the full URL, 
  "http://www.w3schools.com/images/smile.gif"
Often the meta element is used to provide information that is relevant to browsers or search engines like describing the content of your document.

**Page description** (used by search engines)

```html
<meta name="description" content="Free Web tutorials on HTML, CSS, XML, and XHTML"/>
```

**Keywords** (used by search engines)

```html
<meta name="keywords" content="HTML, DHTML, CSS, XML, XHTML, JavaScript, VBScript"/>
```

**Page refresh** (used by browser – returned in the header)

```html
<meta http-equiv="refresh" content="5" />
```

**Page redirect** (used by browser)

```html
<meta http-equiv="refresh" content="5; URL=http://www.unibz.it"/>
```
HTML validation

- You could validate your page - check that they respect the DOCUMENT type that you selected

Validate www.unibz.it
Validate our course web page

http://validator.w3.org/
I've just deleted a blank space
Content vs. Presentation

- HTML tags were originally designed to define the **content** of a document.
- They were supposed to say "This is a header", "This is a paragraph", by using tags like `<h1>`, `<p>`, ...
- The layout was supposed to be taken care of by the browser.
- Netscape and Internet Explorer continued to add new HTML tags and attributes (like the `<font>` tag and the color attribute) to the original HTML specification.
  - Difficult to create Web sites where the **content** of HTML documents was clearly separated from the document's **presentation** layout.
- The World Wide Web Consortium (W3C) created STYLES in addition to HTML 4.0.
Cascading Style Sheets

- **CSS** stands for **Cascading Style Sheets**
- Styles define **how to display** HTML elements
  - Example (inside an element): `<p style="color: sienna; margin-left: 20px"> This is a paragraph </p>`
- Styles are normally stored in external CSS **Style Sheets**
- Styles were added to HTML 4.0 **to solve the problem of mixed content and presentation instructions in one file**
- **External Style Sheets** can save you a lot of work:
  - You can define a style for each HTML element and apply it to as many Web page
- Multiple style definitions will **cascade** into one.
Styles

- When a browser reads a style sheet, it will format the document according to it
- There are three ways of inserting a style sheet:
  - **External Style Sheet**
    ```html
    <head>
    <link rel="stylesheet" type="text/css" href="mystyle.css">
    </head>
    ```
  - **Internal Style Sheet**
    ```html
    <head>
    <style type="text/css">
    body {background-color: red} p {margin-left: 20px}
    </style>
    </head>
    ```
  - **Inline Styles**
    ```html
    <p style="color: red; margin-left: 20px">
    This is a paragraph
    </p>
    ```
Cascading Order

- What style will be used when there is more than one style specified for an HTML element?

- All the styles will "cascade" into a new "virtual" style sheet by the following rules, where number four has the highest priority:
  1. **Browser** default
  2. **External** style sheet
  3. **Internal** style sheet (inside the `<head>` tag)
  4. **Inline** style (inside an HTML element)

- Example: an **inline** style (inside an HTML element) will override a style declared inside the `<head>` tag (**internal**).
CSS Syntax

- The CSS syntax is made up of three parts:
  - selector {property: value}
  - Example: p {font-family: "sans serif"}
- selector = tag name
- property = attribute you want to style
- value = the value you want to assign
- Assign more properties simultaneously: p { text-align: center; color: black; font-family: arial }
- Assign the same property to more tags: h1, h2, h3, h4, h5, h6 {color: green}

put quotes if the value is multiple words
With the class selector you can define **different styles for the same** type of HTML **element**

Style definition example

```
p.right {text-align: right}
p.center {text-align: center}
```

**Usage**

```html
<p class="right"> This paragraph will be right-aligned. </p>
<p class="center"> This paragraph will be center-aligned. </p>
```

More than one class per element is possible

```html
<p class="center bold"> This is a paragraph. </p>
```

A class can be defined for all tags

```html
.center {text-align: center}
```
External vs. Internal

- **External**: this must go in the HTML doc
  <head>
  <link rel="stylesheet" type="text/css"
       href="mystyle.css" />
  </head>
  
  And in the mystyle.css file can go:
  
  hr {color: sienna}
  p {margin-left: 20px}
  body {background-image: url("images/back40.gif")}

- **Internal**
  
  <head>
  <style type="text/css">
  hr {color: sienna} p {margin-left: 20px} body {background-image: url("images/back40.gif")}
  </style>
  
  </head>
Box Model

- All HTML elements can be considered as boxes

```html
div.ex
{
    width:220px;
    padding:10px;
    border:5px solid gray;
    margin:0px;
}
```

eexample
Examples

- **Set the background color**
  This example demonstrates how to set the background color for an element

- **Set the color of the text**
  This example demonstrates how to set the color of the text

- **Align the text**
  This example demonstrates how to align the text

- **Decorate the text**
  This example demonstrates how to add decoration to text

- **Set the font of a text**
  This example demonstrates how to set a font of a text

- **Set the boldness of the font**
  This example demonstrates how to set the boldness of a font

- **All the margin properties in one declaration**
  This example demonstrates how to set a shorthand property for setting all of the margin properties in one declaration

- **The different list-item markers in unordered lists**
  This example demonstrates the different list-item markers in CSS

- **Set the layout of a table**
  This example demonstrates how to set the layout of a table