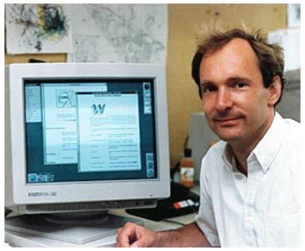
COMP 2021

Unix and Script Programming

Web Programming Fundamentals

The World Wide Web & Internet

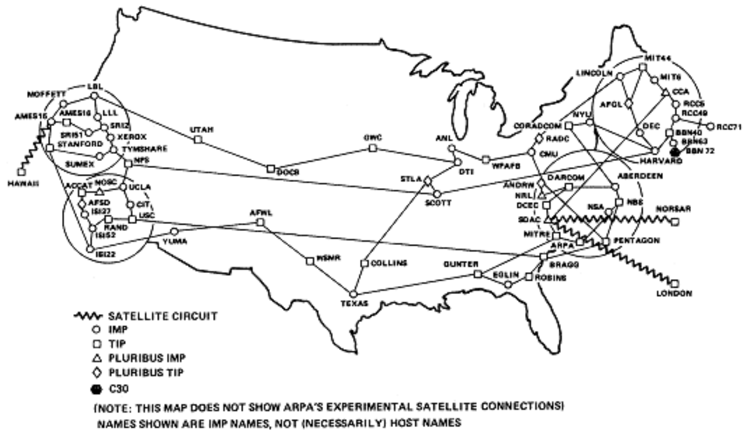
- Internet
  - Originated from ARPANET, was in more recognizable form in 1990
  - A collection of computers or networking devices connected together.
  - Have communication between each other.
  - Decentralized design: no centralized body controls how the Internet functions.



- World Wide Web (WWW)
  - Proposed by Tim Berners-Lee at CERN on 1991.
  - A collection of documents that are interconnected by hyper-links
  - Stored on networked computers over the world
  - These documents are accessed by web browsers and provided by web servers.

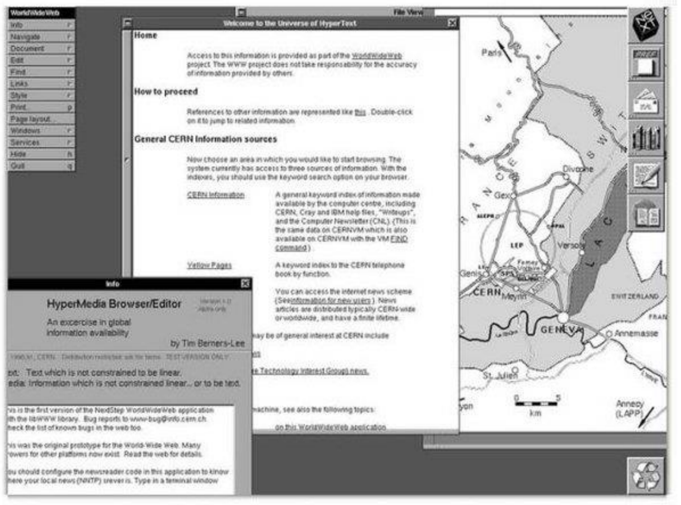
1980 vs. 2014

ARPANET GEOGRAPHIC MAP, OCTOBER 1980



SATELLITE CIRCUIT
○ IMP
□ TIP
△ PLURIBUS IMP
◇ PLURIBUS TIP
● C30

(NOTE: THIS MAP DOES NOT SHOW ARPA'S EXPERIMENTAL SATELLITE CONNECTIONS)
NAMES SHOWN ARE IMP NAMES, NOT (NECESSARILY) HOST NAMES



Home

Access to this information is provided as part of the WorldWideWeb project. The WWW project does not take responsability for the accuracy of information provided by others.

How to proceed

References to other information are represented like this . Double-click on it to jump to related information.

General CERN Information sources

Now choose an area in which you would like to start browsing. The system currently has access to three sources of information. With the indexes, you should use the keyword search option on your browser.

CERN Information	A general keyword index of information made available by the computer centre, including CERN, Cray and IBM help files, "Writeups", and the Computer Newsletter (CNL). (This is the same data on CERNVM which is also available on CERNVM with the VM FIND command).
Yellow Pages	A keyword index to the CERN telephone book by function.
	You can access the internet news scheme (See information for new users). News articles are distributed typically CERN-wide or worldwide, and have a finite lifetime.

ay be of general interest at CERN include

e Technology Interest Group) news.

achine, see also the following topics:

on this WorldWideWeb application

HyperMedia Browser/Editor

An excercise in global information availability

by Tim Berners-Lee

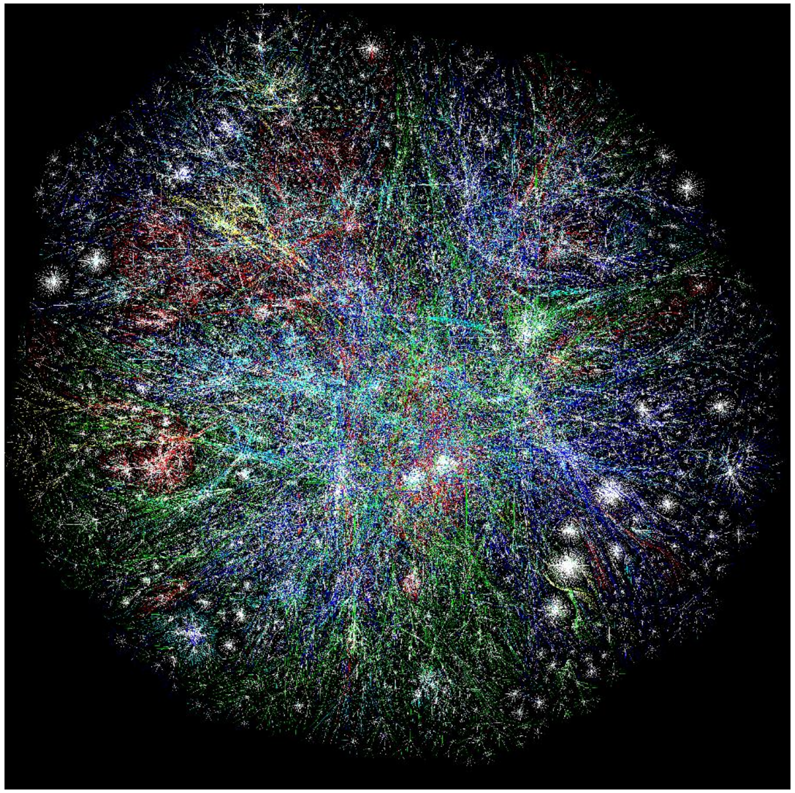
1990,91, CERN. Distribution restricted: ask for terms. TEST VERSION ONLY

xt: Text which is not constrained to be linear.
edia: Information which is not constrained linear... or to be text.

is is the first version of the NextStep WorldWideWeb application ith the libWWW library. Bug reports to www-bug@info.cern.ch heck the list of known bugs in the web too.

is was the original prototype for the World-Wide Web. Many owers for other platforms now exist. Read the web for details.

ou should configure the newsreader code in this application to know here your local news (NNTP) srever is. Type in a terminal window

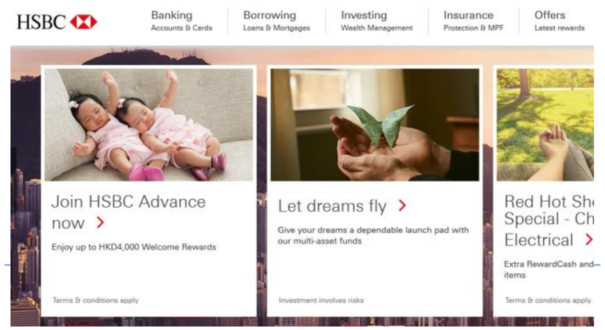


World Wide Web Consortium (W3C)

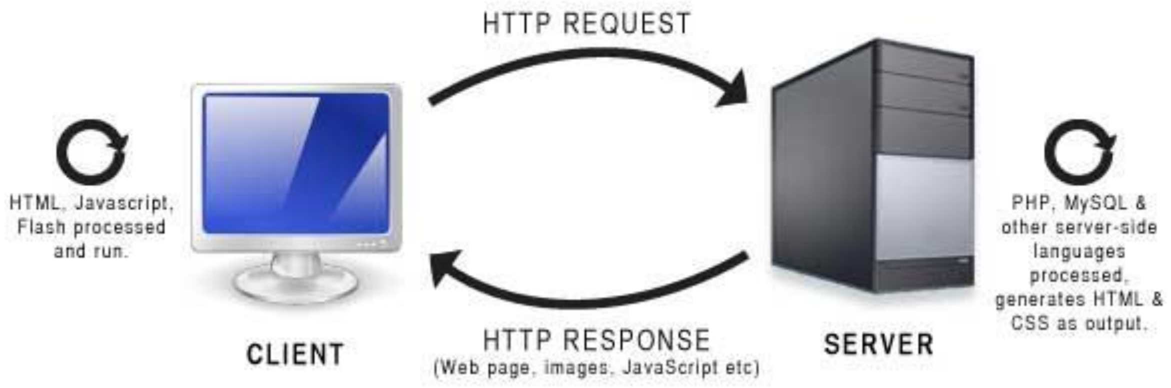
- Web standards are defined by W3C
- The specifications form the Web standards: HTML, CSS, XML, XHTML, …
- W3C's long term goals for the Web are:
  1. *Universal Access:* To make the Web accessible to all by promoting technologies that take into account the vast differences in culture, languages, education, ability, material resources, and physical limitations of users on all continents;
  2. *Semantic Web :* To develop a software environment that permits each user to make the best use of the resources available on the Web;
  3. *Web of Trust :* To guide the Web's development with careful consideration for the novel legal, commercial, and social issues raised by this technology.

The web as a platform for applications

Feature	Web app.	Desktop app.
Graphics	Strong	Unlimited
User interaction	Strong	Unlimited
Network usage	High	Varies
Accessible from	Any computer	Where installed
Upgrade cost	Update servers	Update desktop
Data backup cost	Backup servers	Backup desktop
Popularity	Increasing	Dominant

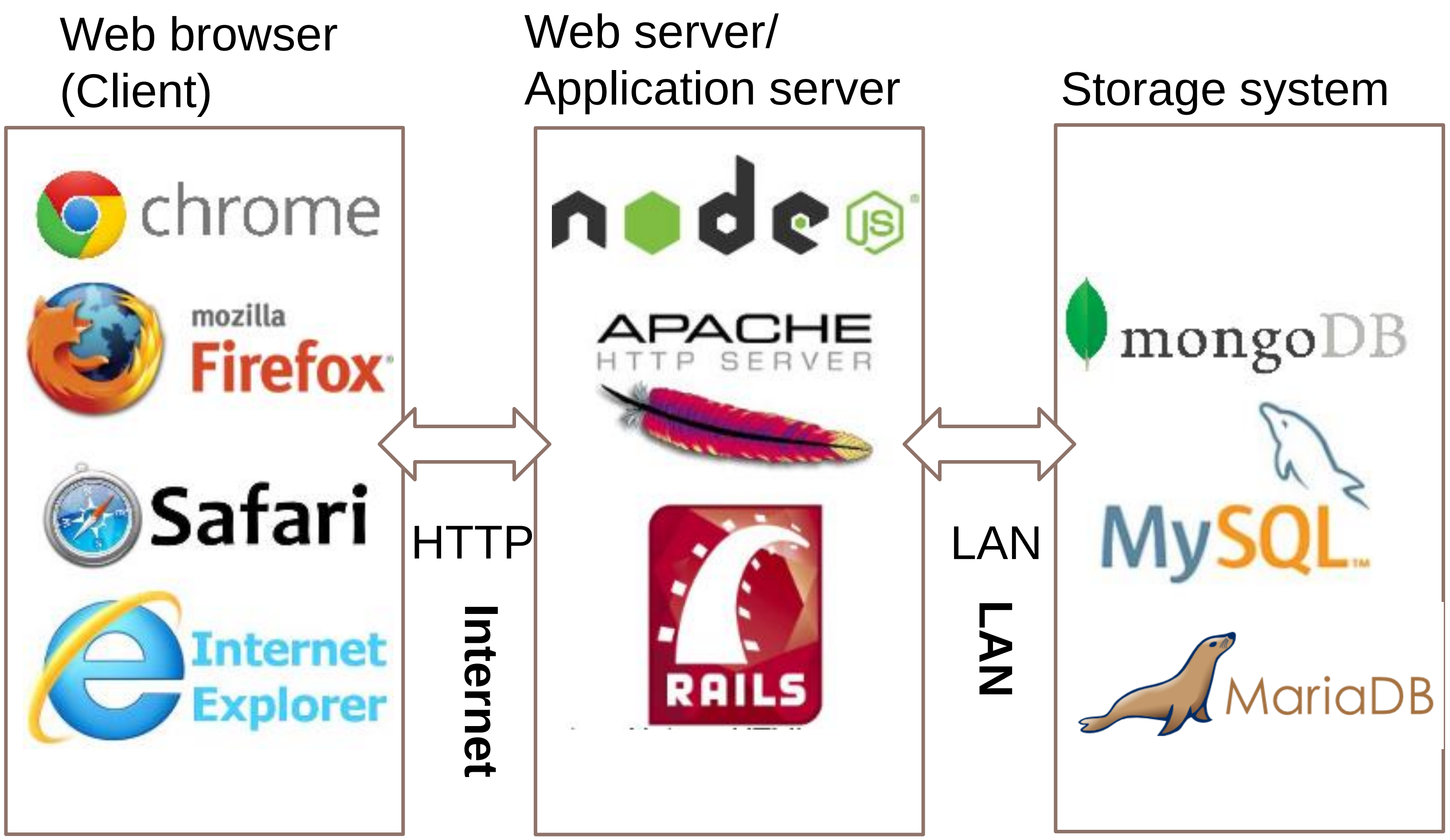


Client and Server


HTTP REQUEST
HTML, Javascript, Flash processed and run.
CLIENT
HTTP RESPONSE
(Web page, images, JavaScript etc)
SERVER
PHP, MySQL & other server-side languages processed, generates HTML & CSS as output.

- Client
  - Reads in the homepage (HTML page), parse it and display it with appropriate layout
- Server
  - receives requests from client computers, processes and sends the output.
  - provides static / dynamic webpages (e.g. retrieve and store information from/to the database and generate dynamic homepage to web clients)
- Web Development / Web Programming
  - The process of creating, modifying web pages.

Web Application Architecture



Web Programming

- Write programs to enable interactions between web clients and web servers
- To be more concrete, your programs are responsible for:
  - deciding the layout of the page in advance
  - creating web pages on-the-fly in responding users' input
  - recording/retrieving users' information to/from the database
- Client-side and server-side scripting

Client-side scripting

- Usually embedded into HTML pages
- User's web browser executes the script, then displays the document, including any visible output from the script.
- Script language: JavaScript, Ajax, ActionScript (animation for Adobe Flash Player)

- Function: add interactivity to HTML pages
  - Put dynamic text into an HTML page
  - Validate data before it is submitted to a server
  - Response to certain user actions, (e.g., clicking a button)
  - Create cookies

Server-side scripting

- Executed at web server to produce a response customized for each user's (client's) request to the website
- In the early days, a combination of C, Perl scripts and shell scripts using the Common Gateway Interface (CGI)
  - Executed by the operating system, and results served back by the web server
- Many modern web servers can directly execute on-line scripting languages
  - PHP: Open source, strong database support (*.php)
  - ASP.NET: Microsoft product, uses .Net framework (*.asp)
  - JSP: Java via JavaServer Pages (*.jsp)
  - Ruby (*.rb), Python (*.py), ColdFusion Markup (*.cfm), Perl via CGI.pm module (*.cgi, *.pl)
- …

Client-side: Hypertext Markup Language (HTML)

Browser environment is different

- Traditional app: GUIs based on pixels
  - Since 1970s: software accessed mapped framebuffers (R/G/B)
  - Toolkits build higher level GUI widgets (buttons, tables, etc.)
  - Until the most recent HTML5's canvas region, you couldn't write pixels
- Web browsers display documents described in HTML
  - Only give the content and structure of the document, leave visualization to the browser
    - Browsers vary (graphical, text based, mobile devices)
    - User preferences vary (some people like larger fonts)
    - Environment varies (screen sizes, fonts available, etc.)
  - But authors want to control what the document looks like
- Trend towards separating content from presentation
  - Cascading Style Sheets (CSS)

HTML: HyperText Markup Language

- It is not a programming language.
  - Cannot be used to describe computations.
  - Use to describe the general form and layout of documents to be displayed by the browser.
- Markup language: include "Content" and "Directives" (i.e. control)
  - Example: <i>italics word</i>, <title>Title words</title>
- Approach
  - Start with content to be displayed
  - Annotate it with <> tags

HTML Tags

- Tags can provide:
  - Formatting information (e.g. <i> for italic)
  - Meaning of text:
    - <h1> means top-level heading
    - <p> means paragraph
    - <ul> <li> for unordered (bulleted) list
  - Additional information to display (e.g. <img>)
- Tags can have tags inside (nesting supported)

Example of HTML – Start with raw context text

Introduction

There are several good reasons for taking COMP2021: Unix and Script Programming

You will learn a variety of interesting concepts.

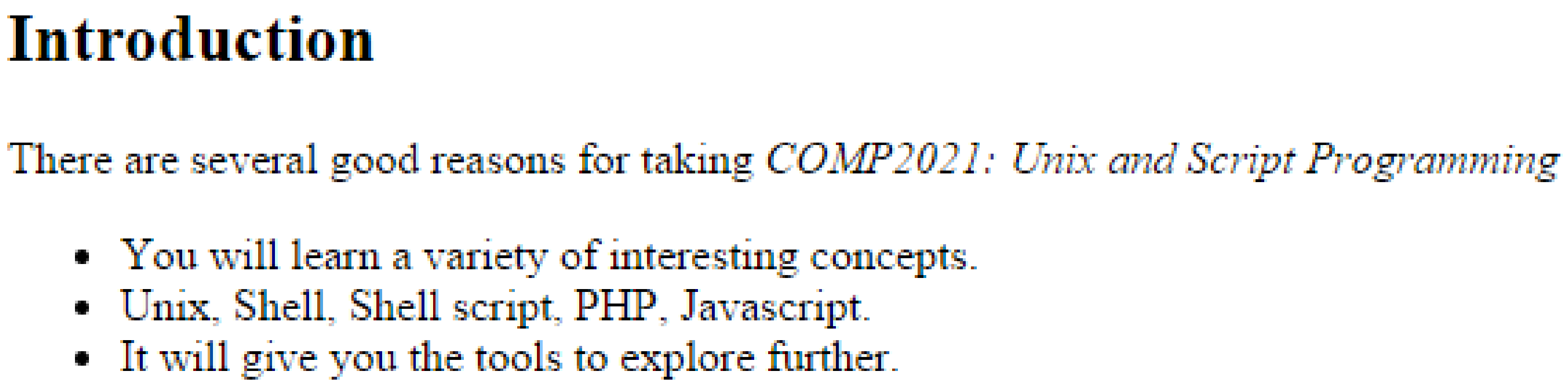
Unix, Shell, Shell script, PHP, JavaScript.

It will give you the tools to explore further.

Example of HTML - Annotate with tags

```
<h2>Introduction</h2>
<p>
   There are several good reasons for taking
    <i>COMP2021: Unix and Script Programming</i>
</p>
<ul>
    <li> You will learn a variety of interesting concepts. </li>
    <li> Unix, Shell, Shell script, PHP, JavaScript. </li>
    <li> It will give you the tools to explore further.  </li>
</ul>
```

Example of HTML – Browser Output



Introduction

There are several good reasons for taking *COMP2021: Unix and Script Programming*

- You will learn a variety of interesting concepts.
- Unix, Shell, Shell script, PHP, Javascript.
- It will give you the tools to explore further.

HTML Evolution

- Browser implementation quirks
  - What to do if you see " <p> Some text" (missing closing </p>)?
  - 1. Complain bitterly about malformed HTML.
  - 2. Figure out there was a missing </p> , add it, and continue processing.
- Forked into HTML and XHTML (XML-based HTML)
  - XHTML is more strict about adhering to proper syntax XHTML to add structure, conventions – early 2000's;
  - Now moving to HTML5
- Cascading Stylesheets (CSS): 1996; current implementation CSS3
- Javascript (1995), Flash (1996), AJAX, JQuery

XHTML

- Document Structure
  - XHTML DOCTYPE is **mandatory**
  - The xmlns attribute in <html> is **mandatory**
  - <html>, <head>, <title>, and <body> are **mandatory**
- XHTML Elements
  - XHTML elements must be **properly nested**
  - XHTML elements must always be **closed**
  - XHTML elements must be in **lowercase**
  - XHTML documents must have **one root element**
- XHTML Attributes
  - Attribute names must be in **lower case**
  - Attribute values must be **quoted**
  - Attribute minimization is **forbidden**

XHTML with minimum required tags

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Indicate that this is an XHTML document, conforming to version 1.0 of the standard; use these lines verbatim in all the web pages you create for this class.

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

Outermost element containing the document

```
<head>
  <title>Title of document</title>
</head>
```

Contains miscellaneous things such as page title, CSS stylesheets, etc.

```
<body>
  some content
</body>
```

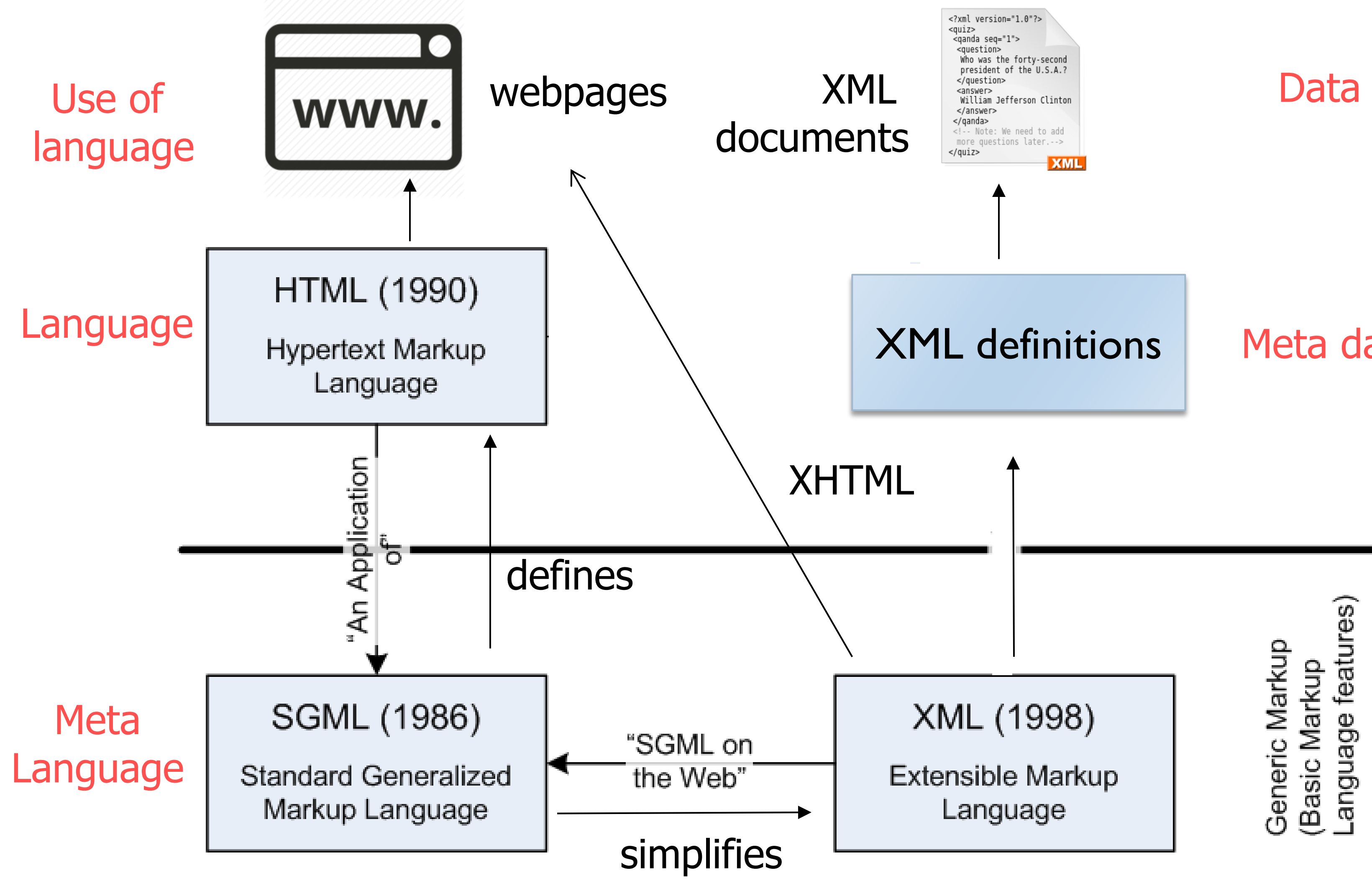
The main body of the document

```
</html>
```

HTML vs. XHTML

- HTML supports the same tags, same features, but allows quirkier syntax
  - Can skip some end tags, such as </br>, </p>
  - Not all attributes have to have values: <select multiple>
  - Elements can overlap: <p><b> first </p><p>second</b>third</p>
- Early browsers tried to "do the right thing" even in the face of incorrect HTML
  - Ignore unknown tags
  - Carry on even with obvious syntax errors such as missing <body> or </html>
  - Infer the position of missing close tags
  - Guess that some < characters are literal, as in "What if x < 0?"
  - Not obvious how to interpret some documents (and browsers differed)

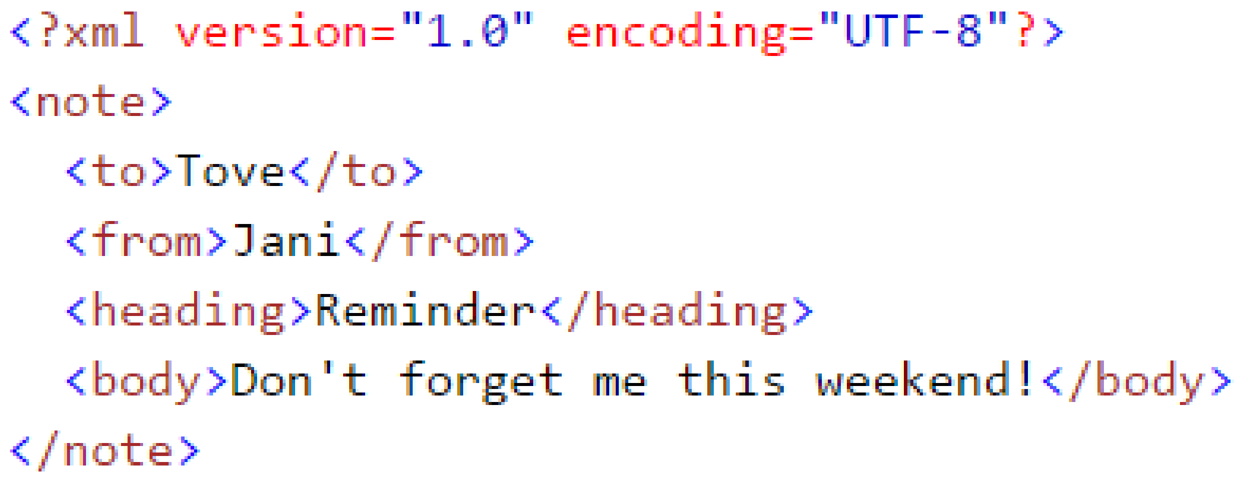
SGML, HTML and XML



```
<?xml version="1.0"?>
<quiz>
 <qanda seq="1">
  <question>
   Who was the forty-second
   president of the U.S.A.?
  </question>
  <answer>
   William Jefferson Clinton
  </answer>
 </qanda>
 <!-- Note: We need to add
  more questions later.-->
</quiz>
```

XML

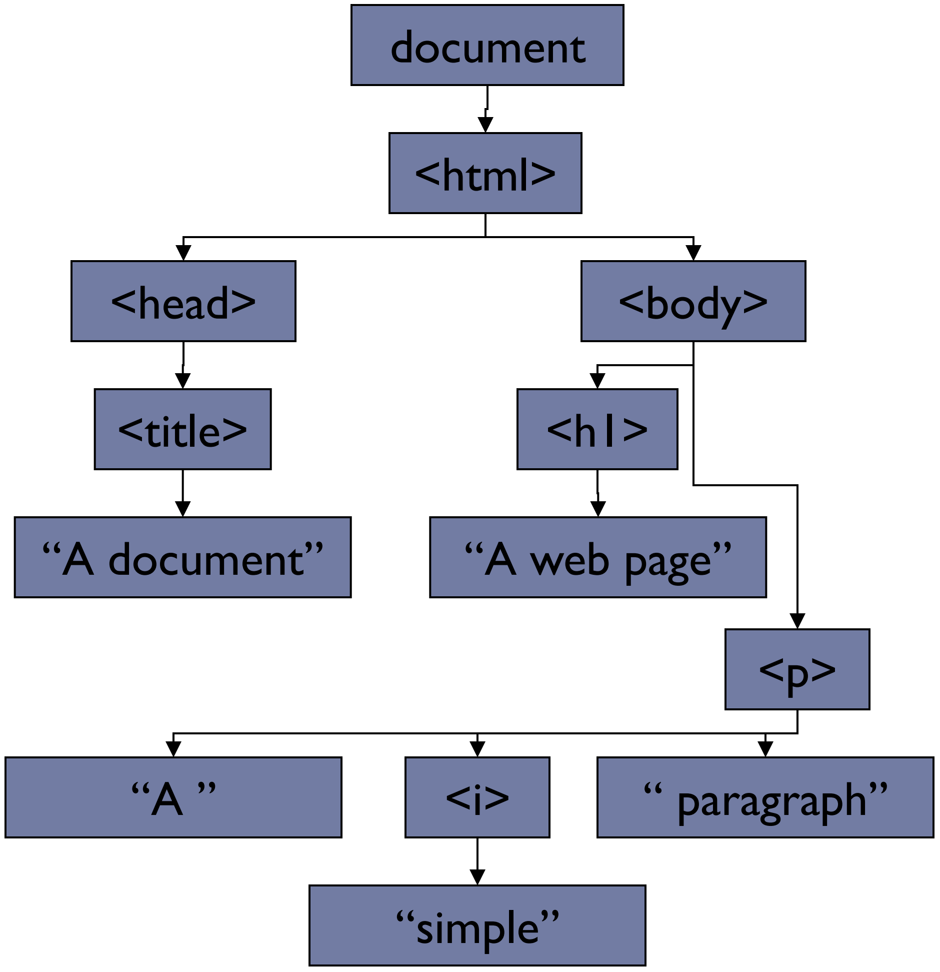
- XML stands for E**X**tensible **M**arkup **L**anguage.
- XML was designed to store and transport data.
- XML was designed to be both human- and machine-readable.
- Example:



```xml
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

The document as a tree

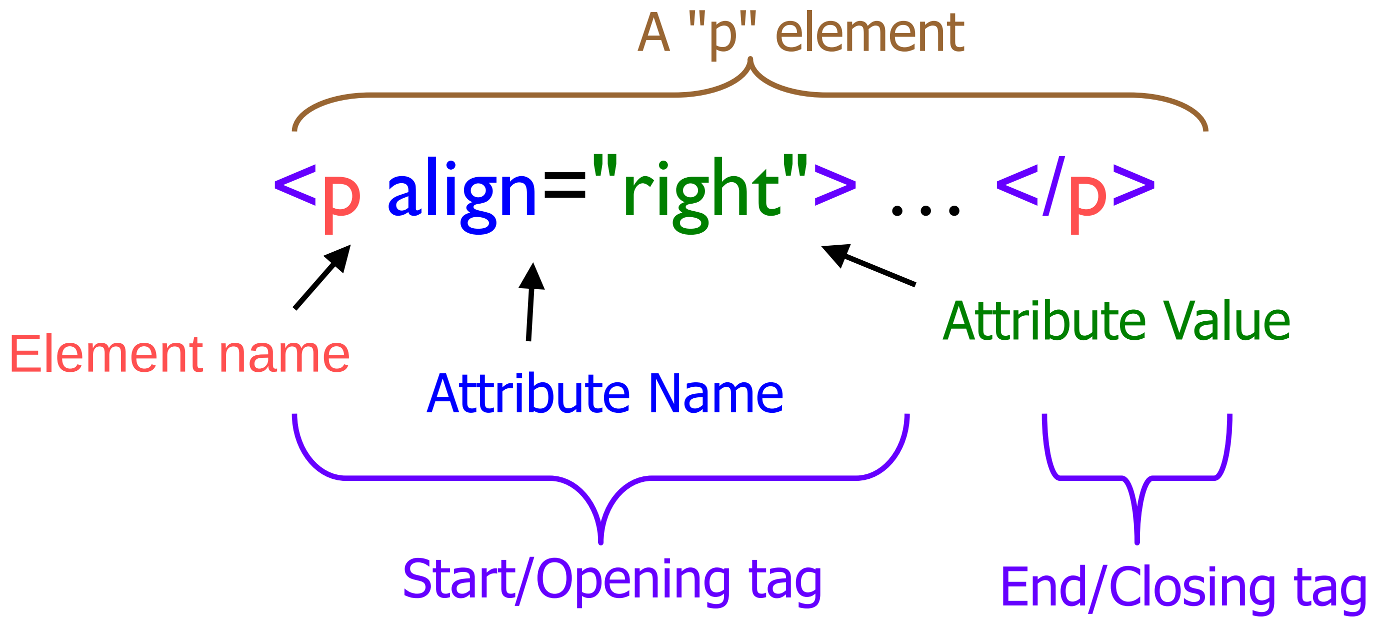
```
<html>
 <head>
  <title>A Document</title>
 </head>
 <body>
  <h1>A web page</h1>
  <p>A <i>simple</i> paragraph</p>
 </body>
</html>
```



(X)HTML Quick Reference (Self-study)

Elements, Tags, Attributes

- An element in a predefined building block of the document.
- An element is marked using tags in the document as



- An element has a name.
- An element may have zero or more attributes (some are required and some are optional)

Main HTML Elements

- **`DOCTYPE`**
  - Specify which "version" of HTML/XHTML the current document adheres to
- **`html`**
  - Appear exactly once.
- **`head`**
  - **`title`** element required
  - Optional elements:
    - **`base, meta, script, style, link`**
  - Appear exactly once
- **`body`**
  - Appear exactly once (immediately after the **`head`** element)

HTML Character Entity References

- Some characters have special meaning in an HTML documents and therefore must be represented as character entity references.
- A character entity reference can take two forms
  - &name; name is a predefined name
  - &#N; where N is an integer number

Result	Description	Entity Name	Entity Number
	non-breaking space	&nbsp;	&#160;
<	less than	&lt;	&#60;
>	greater than	&gt;	&#62;
&	ampersand	&amp;	&#34;
"	quotation mark	&quot;	&#38;
'	apostrophe	&apos;	&#39;

Reference: http://www.w3schools.com/html/html_entitiesref.asp

White Space

- Each newline/tab character is replaced by a space character.
- Consecutive white space characters, including tab, newline, and space characters, are collapsed into a single space character.
- To output multiple spaces, use non-breaking space entity (&nbsp;) . For example,

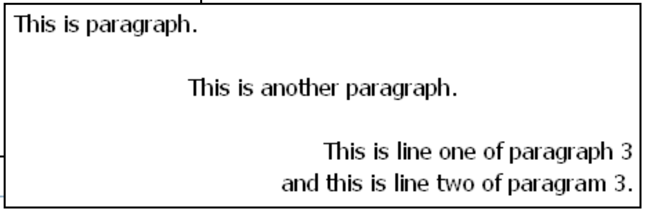
  ```
  A&nbsp;&nbsp;&nbsp;B
  ```

  will produce three spaces between A and B
- Exception: white space characters are preserved in the <pre> element.

Content Formatting

- Headings (h1, h2, h3, h4, h5, h6)
  - <h1> defines the largest heading. <h6> defines the smallest heading.
- Paragraphs (p)
  - May have attribute align with possible value of "left", "right", "center", and "justify".
- Line break (br)
  - Used when you want to end a line, but don't want to start a new paragraph. The <br> tag forces a line break wherever you place it.

```
<p>This is paragraph.</p>
<p align="center">This is another
paragraph.</p>
<p align="right">This is line one of
paragraph 3<br/>
and this is line two of paragram
3.</p>
```



```
This is paragraph.

                    This is another paragraph.

                              This is line one of paragraph 3
                           and this is line two of paragram 3.
```

Text Formatting

- Physical Character Styles
  - Bold (b), italics (i), teletype (tt), underline (u), subscript(sub), superscript (sup), small text (small), big text (big), deleted text (del), inserted text (ins)

- Logical Character Styles
  - Emphasized text (em), strong text (strong), computer code (code), sample computer code text (samp), citation (cite), …

HTML Link and Anchor (a)

- To create a link to a resource identifiable by a URL
  - href: specify a URL of the target resource
  - target: specify where to display the target document
    - e.g.: `<a href="index.htm" target="_blank">Home</a>`
    - Open the document "index.htm" in a new browser window
- Can also be used to create an anchor within a document
  - name: specify the anchor name
    - e.g.: `<a name="chap1"></a><h2>Chapter 1</h2>`

      The above anchor can be referred to in a URL as

      `<a href="http://host/file.html#chap1">Chapter 1</a>`
- **Note**: The role of "anchor" may be replaced by the "id" attribute in the future and any element can be treated as an anchor.

Absolute and Relative URLs

- Absolute URL
  - A complete URL beginning with http://
  - e.g., **`http://www.example.com/foo/index.html`**
- Relative URL
  - Interpreted as relative to the URL of the current document
  - Suppose the URL of the current document is
    `http://www.example.com/foo/bar/index.html`

    `path/index.html` is interpreted as
    `http://www.example.com/foo/bar/path/index.html`

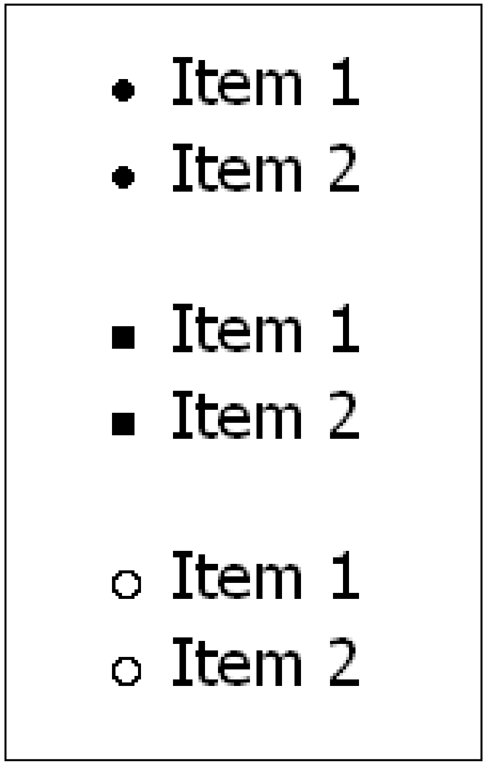
    `/index.html` is interpreted as
    `http://www.example.com/index.html`

    `../index.html` is interpreted as
    `http://www.example.com/foo/index.html`

Unordered List (`ul`)

- Use <li> to specify list items

```
<ul>
<li>Item 1</li>
<li>Item 2</li>
</ul>

<ul type="square">
<li>Item 1</li>
<li>Item 2</li>
</ul>

<ul type="circle">
<li>Item 1</li>
<li>Item 2</li>
</ul>
```



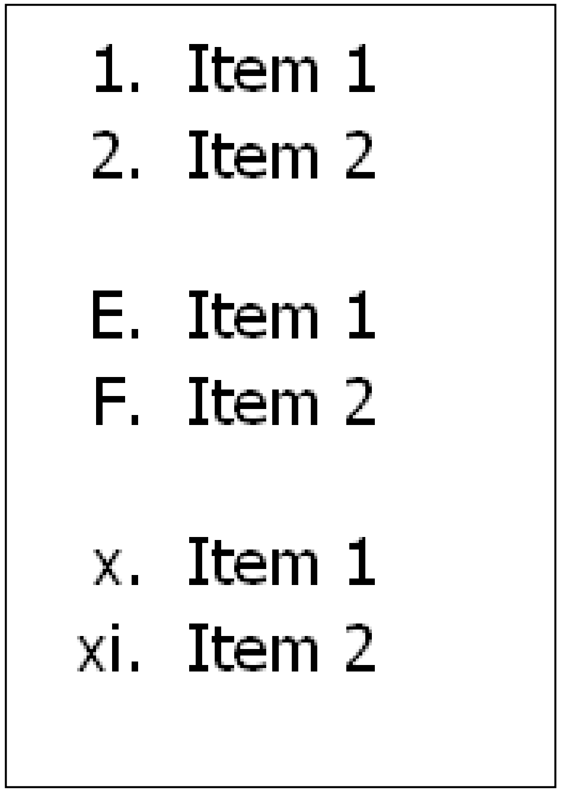
- Item 1
- Item 2

▪ Item 1
▪ Item 2

○ Item 1
○ Item 2

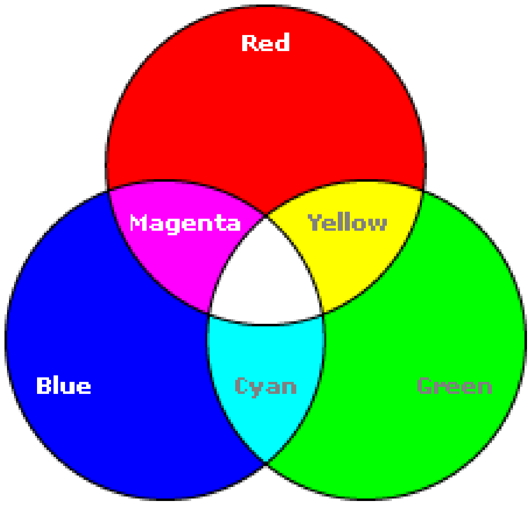
Ordered List (`ol`)

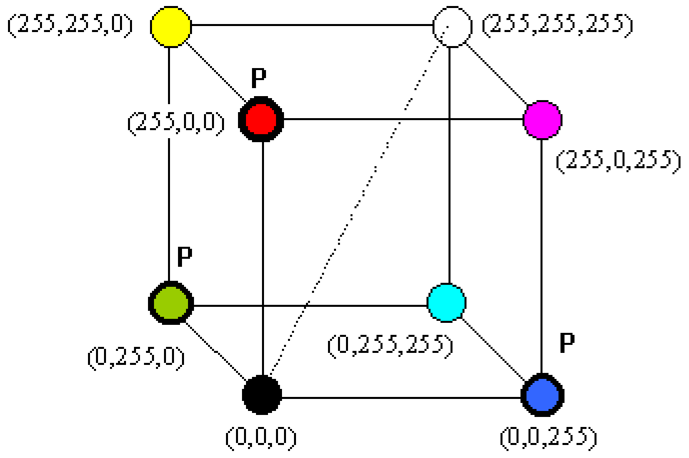
```
<ol>
<li>Item 1</li>
<li>Item 2</li>
</ol>

<ol type="A" start="5">
<li>Item 1</li>
<li>Item 2</li>
</ol>

<ol type="i" start="10">
<li>Item 1</li>
<li>Item 2</li>
</ol>
```

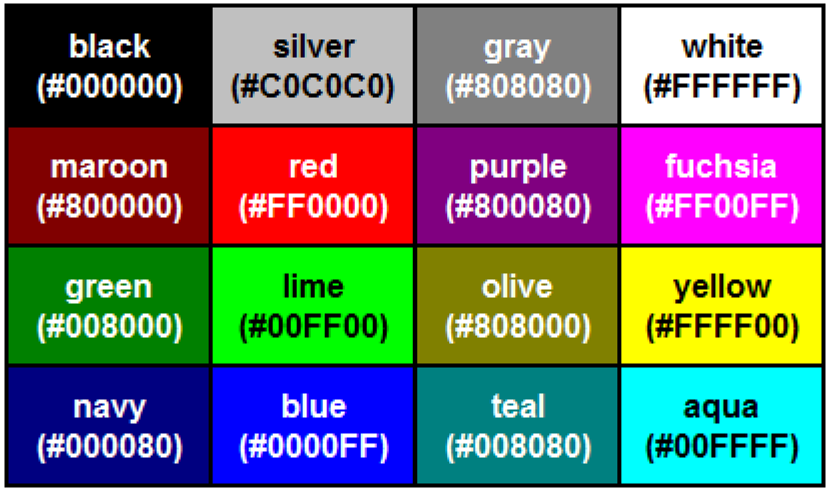


```
 1. Item 1
 2. Item 2

 E. Item 1
 F. Item 2

 x. Item 1
xi. Item 2
```

RGB color model





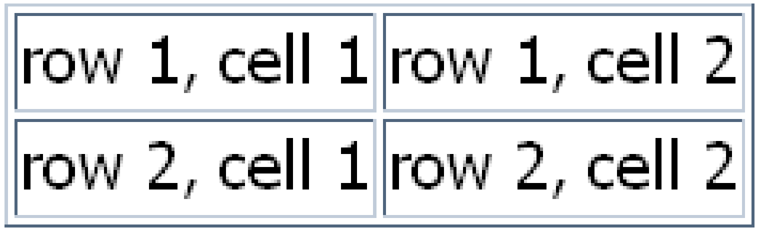


black (#000000)	silver (#C0C0C0)	gray (#808080)	white (#FFFFFF)
maroon (#800000)	red (#FF0000)	purple (#800080)	fuchsia (#FF00FF)
green (#008000)	lime (#00FF00)	olive (#808000)	yellow (#FFFF00)
navy (#000080)	blue (#0000FF)	teal (#008080)	aqua (#00FFFF)

Tables (table)

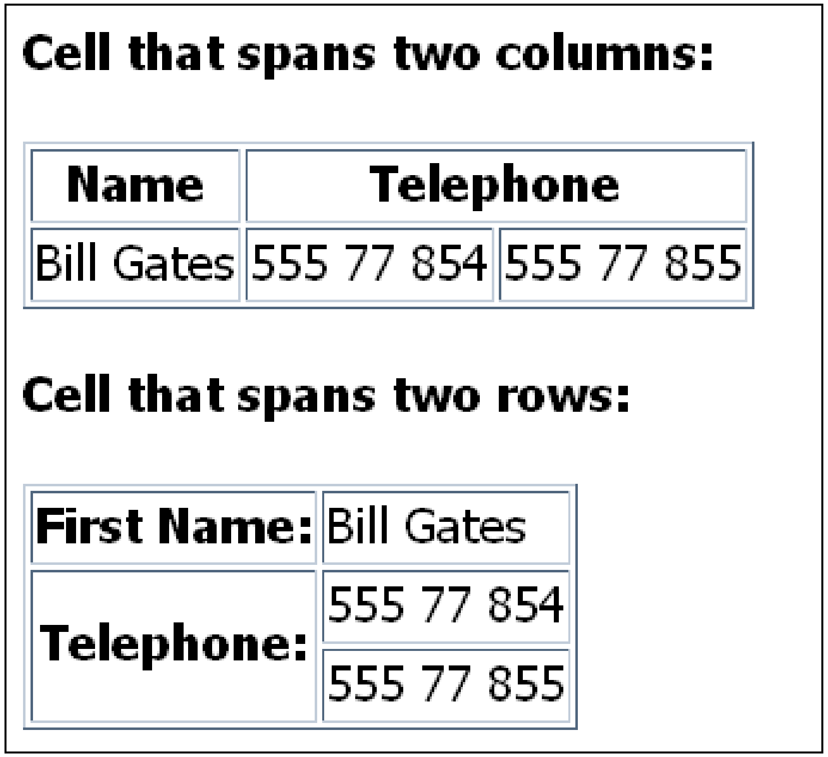
- Define a table
- A table is divided into rows (with the <tr> tag), and each row is divided into data cells (with the <td> tag)

```
<table border="1">
<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>
```



row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

```
<h4>Cell that spans two
columns:</h4>
<table border="1">
<tr>
  <th>Name</th>
  <th colspan="2">Telephone</th>
</tr>
<tr>
  <td>Bill Gates</td>
  <td>555 77 854</td>
  <td>555 77 855</td>
</tr>
</table>

<h4>Cell that spans two rows:</h4>
<table border="1">
<tr>
  <th>First Name:</th>
  <td>Bill Gates</td>
</tr>
<tr>
  <th rowspan="2">Telephone:</th>
  <td>555 77 854</td>
</tr>
<tr>
  <td>555 77 855</td>
</tr>
</table>
```



Cell that spans two columns:

Name	Telephone	
Bill Gates	555 77 854	555 77 855

Cell that spans two rows:

First Name:	Bill Gates
Telephone:	555 77 854
	555 77 855

Embedded Images (`img`)

- To embed an image (jpg, gif, png) in a document.
- Example

```
<img src="SomeFile.gif" alt="My Dog" title="My
  Dog" width="400" height="300" />
```

- Basic attributes:
  - `src`: URL of the image (required)
  - `alt:` Alternate text description of the image (technically required)
  - `title:` text to appear when mouse cursor hovers above the image
  - `width, height`: display the image in this dimension

Block-level and Inline Elements

- Block-level Elements
  - A block-level element takes up the full width available, with a new line before and after
  - e.g.: p, div, table, list, h1, …, h6, …
  - **`div`** is a generic block element

- Inline Elements
  - An inline element takes up only as much width as it needs, and does not force new lines
  - e.g.: img, a, b, i, button, span, …
  - **`span`** is a generic inline element

- Note: The "block/inline" property can be modified using CSS.

Client-side: Cascading Style Sheets (CSS)

Why CSS?

- How what font type and size does <h1> Introduction </h1> generate?
  - Answer: Some default from the browser (HTML tells **what** browser **how**)
- Early HTML - Override defaults with attributes
  - <table border = "2" bordercolor = "black">
- Style sheets were added to address this:
  - Specify style to use rather than browser default
  - Not have to code styling on every element

Key concept: Separate style from content

- Content (what to display) is in HTML files
- Formatting information (how to display it) is in separate style sheets (.css files).
- Use an element attribute named class to link (e.g. <span class=”test”>)
- Result: define style information once, use in many places
  - Consider can you make all the text in the app slightly bigger?
  - Or purple is our new company color.

- **DRY principle: Don't Repeat Yourself**

Connecting HTML to CSS

- Styles can be embedded inline with the *style* attribute
- Style sheets can be chosen by media type
  - Simply add a *media* attribute to the *link* or *style tags*
  - Choose from: screen, tty, tv, projection, handheld, braille, aural, all
- HTML document can provide several stylesheet options
  - Give titles to each stylesheet
  - One preferred (default) style, the rest are alternates
    - e.g., http://www.w3.org/Style/Examples/007/alternatives.html
- Default configuration in internal browser stylesheet and user stylesheet

Connecting HTML to CSS (cont.)

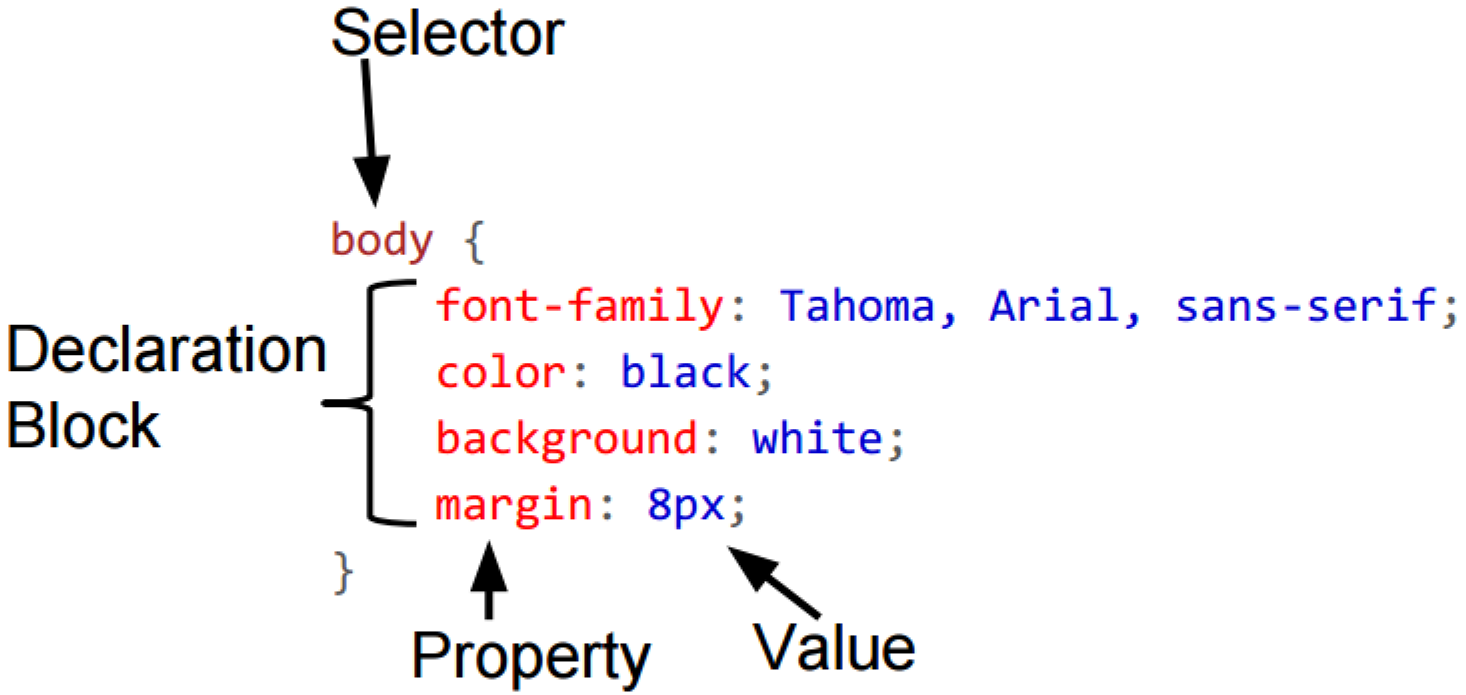
- HTML document typically refers to external style sheet

```
<HEAD>
  <LINK rel="stylesheet" type="text/css"
  href="fluorescent.css">
</HEAD>
```

- Style sheets can be embedded:

```
<HEAD><STYLE type="text/css">
    <!-- …CSS DEFINITIONS.. -->
</STYLE></HEAD>
```

CSS Rules

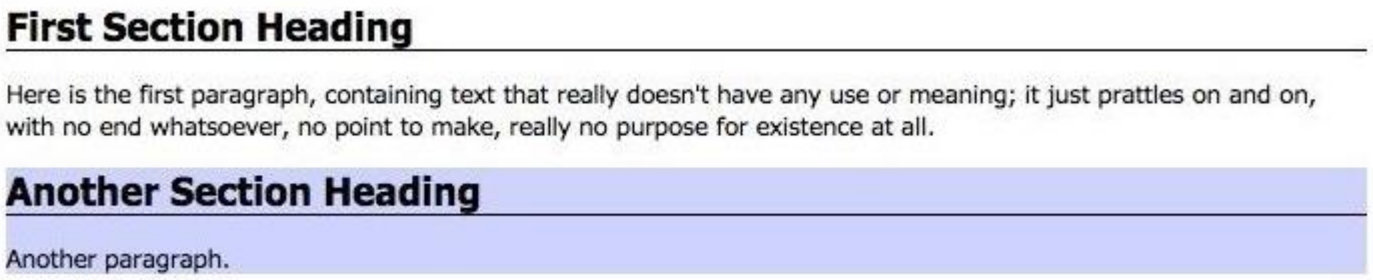


```
body {
    font-family: Tahoma, Arial, sans-serif;
    color: black;
    background: white;
    margin: 8px;
}
```

CSS Example

```css
body {
  font-family: Tahoma, Arial, sans-serif;
  font-size: 13px;
  color: black;
  background: white;
  margin: 8px;
}
h1 {
  font-size: 19px;
  margin-top: 0px;
  margin-bottom: 5px;
  border-bottom: 1px solid black
}
.shaded {
  background: #d0d0ff;
}
```

```html
<body>
  <h1>First Section Heading</h1>
  <p>
    Here is the first paragraph, containing
    text that really doesn't have any use
    or meaning; it just prattles on and on,
    with no end whatsoever, no point to
    make, really no purpose for existence
    at all.
  </p>
  <div class="shaded">
    <h1>Another Section Heading</h1>
    <p>
      Another paragraph.
    </p>
  </div>
</body>
```



First Section Heading

Here is the first paragraph, containing text that really doesn't have any use or meaning; it just prattles on and on, with no end whatsoever, no point to make, really no purpose for existence at all.

Another Section Heading

Another paragraph.

CSS Selectors

CSS Selector	CSS	HTML

Tag name	`h1 {`
	`  color: red;`
	`}`
		`<h1>Today’s Specials</h1>`

Class attribute	`.large {`
	`  font-size: 16pt;`
	`}`
		`<p class="large">...`

Tag and Class	`p.large {...}`	`<p class="large">...`

Class id	`#p20 {`
	`  font-weight: bold;`
	`}`
		`<p id="p20">...`

CSS Pseudo Selectors

- hover - Apply rule when mouse is over element (e.g. tooltip)

```
p:hover,  a:hover {
      background-color: yellow;
 }
```

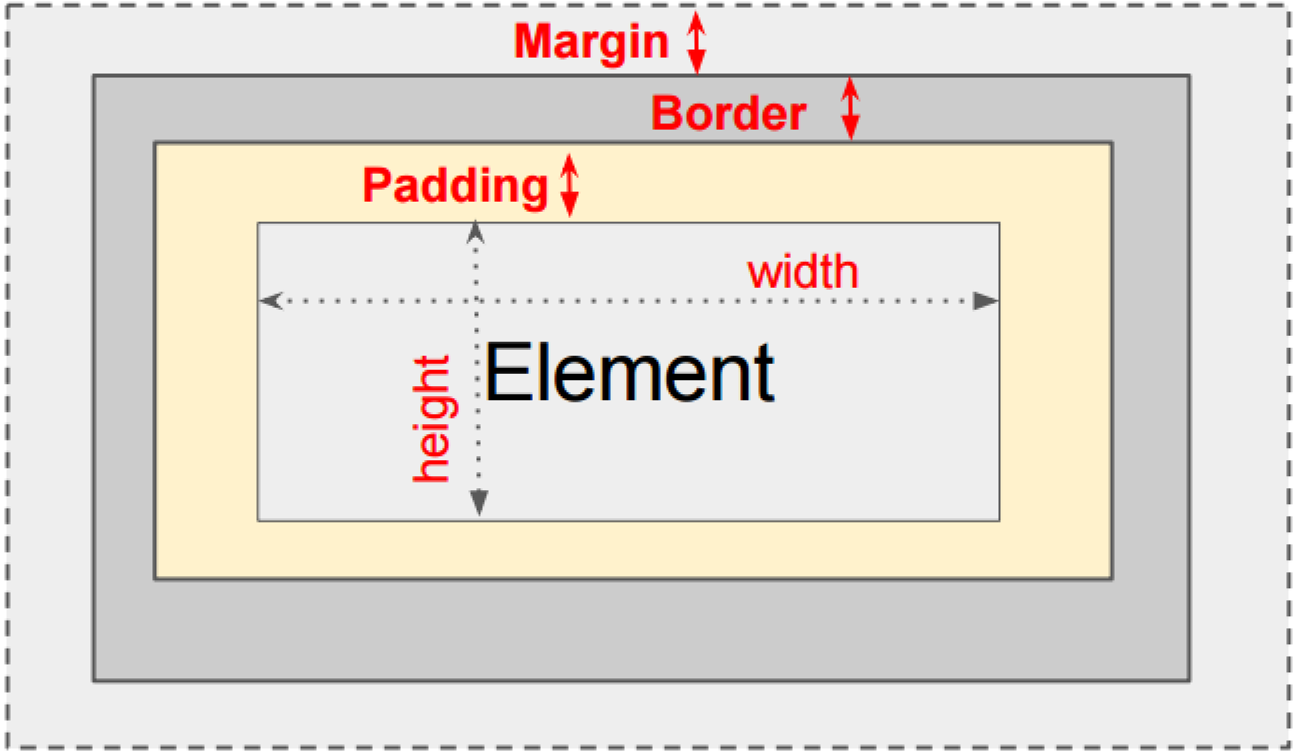
- a:link, a:visited - Apply rule when link has been visited or not visited (link)

```
a:visited { color: green; }
a: link: {color: blue; }
```

CSS Properties

- Control many style properties of an element:
  - Coloring
  - Size
  - Position
  - Visibility
  - Many more

Example: CSS Box Model



Total element width =
width +
left padding +
right padding +
left border +
right border +
left margin +
right margin

Margin & Padding
Transparent