ICT286
Web and Mobile Computing

Topic 1
Introduction to WWW and HTML

Lecture Objectives

- Learn about the concepts that underlie the World Wide Web (WWW).
- Define what the WWW is.
- Understand the basic operations performed by a web server.
- Understand the basic mechanism used by HTTP to communicate.
- Understand what HTML is and how it was developed.
- Understand the basics of the HTML language, except forms and frames.
- Be able to write simple but valid documents in XHTML5.

A Little History - The beginning of the Web

- The World Wide Web was originally conceived as a tool to link information kept by different researchers in CERN, the European Organization for Nuclear Research.

- In 1989, the original proposal for this system was put forward by Tim Berners-Lee.

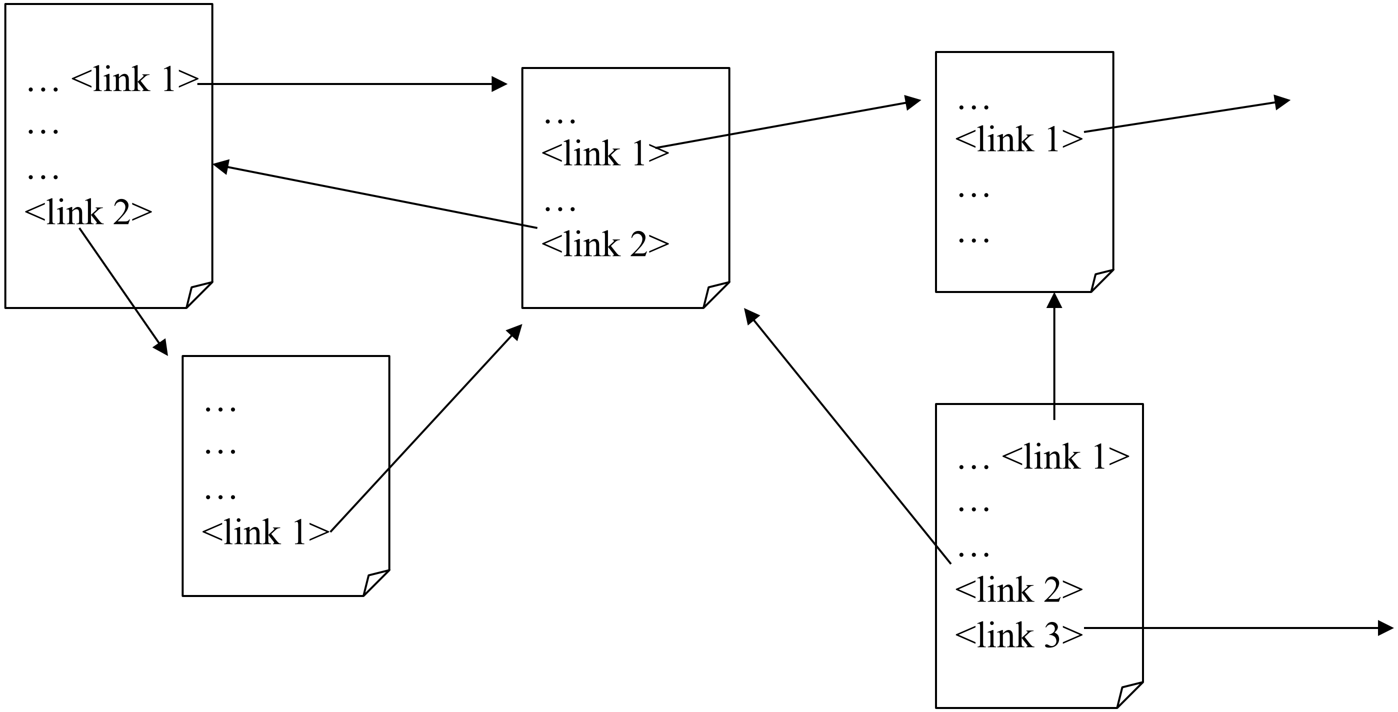
The Original Vision

- Tim Berners-Lee’s original intention was to:
    - have access to information for all
    - use hypertext
    - facilitate collaborations.

- He described the web as a "*distributed heterogeneous collaborative multimedia information system*".

Hypertext

Using links to tie together documents:



History - Graphical Browsers

- The original *browsers* Berners-Lee wrote in 1989 to view these hyperlinked documents were all text-based. He later wrote a graphical one called *worldwideweb*.
- In 1993, the graphical browser Mosaic was released by Marc Andreessen of NCSA (National Centre for Supercomputing Applications) at University of Illinois at Urbana-Champaign.
- In 1994, Andreessen left NCSA and together with Jim Clark formed Netscape Communications. The development of Netscape's browser, Navigator, drove the popularity of the WWW.

History - W3C and Microsoft Internet Explorer

- Also in 1994, CERN and MIT (Massachusetts Institute of Technology), with Berners-Lee leading, formed the World-Wide-Web Consortium W3C, to
    - develop web protocols
    - encourage inter-operability between web sites
- In 1995, Microsoft finally recognised the potential of the Internet and the Web, and shifted it business focus to large scale penetration of the Web.

History - Microsoft, Netscape and W3C

- The development of Microsoft's Internet Explorer, in competition with Netscape's Navigator/Communicator, plus the activities of the W3C, have largely led us to the WWW scene going into the late 1990s.
    - A lot of what we associate with web activities today were defined at that time. eg. the operations of web browsers, the use of HTML tables for image positioning, the widespread adoption of JavaScript, etc.
- It is the WWW, by its nature, a lot more user friendly than the applications that existed on the Internet before it, that led to the world-wide adoption of the Internet as a tool for everyone to use.

WWW – The Concept

- Universal readership
- The use of hypertext
- Allows searching for information
- Based on the client-server model
- Communication by *format negotiation*
  - that is, two parties involved in communication can negotiate between themselves what format they want to communicate in.
  - HTTP supports this.

WWW – The Original Protocols

- Addressing using:
  - Universal Resource Locators (URL)
- Information Exchange using:
  - Hypertext Transfer Protocol (HTTP)
- Formatting using:
  - The Hypertext Markup Language (HTML)

WWW – The Protocols (cont'd)

- Today, there are a lot more protocols that have been developed which people consider to be part of the WWW.
- eg
    - XML (eXtensible Mark-up Language)
    - CSS (Cascading Style Sheets)
    - DOM (Document Object Model)
    - etc.

What is the WWW?

- The World Wide Web Consortium (W3C) describes the web as:
  - The World Wide Web (known as *WWW*, *Web* or *W3*) is the universe of network-accessible information, the embodiment of human knowledge.
  - The Web has a body of software, and a set of protocols and conventions.

- This description basically includes the web in anything W3C chooses to work on.

What is the WWW? (cont'd)

- We can use the following as a *definition* for the World Wide Web:
  - The World Wide Web (WWW) is a global information system where

    1) resources are addressed using URLs, or its subsequent extensions/follow-ons;

    2) Communication occurs using HTTP or its subsequent extensions/follow-ons;

    3) Information is linked using hypertext based on HTML, or its subsequent extensions/follow-ons;

Difference between WWW and *the Internet*

- Note the difference between how we have defined the WWW and the Internet.
    - The Internet is defined using TCP/IP which deals with how to transport packets. It doesn’t mention specific protocols for applications.
    - The WWW defines protocols for applications.
    - So WWW can be considered an application of the Internet.
- Then again, we could conceivably implement the WWW over a non-TCP/IP network, which would make it non-Internet based.

The World Wide Web Consortium (W3C)

- Responsible for standards on the World Wide Web.
    - HTTP (Hypertext Transfer Protocol)
    - HTML (Hypertext Mark-up Language)
    - XML (Extensible Mark-up Language)
    - etc.
- Also develops guidelines for use of web technologies.
- A consortium of many organisations
    - unlike IETF, consortium membership is by payment.

Network Application Components

- Network applications have several components (they are not only a protocol). eg. The Web :
  - A standard for document formats (HTML)
  - Web browsers (Firefox, IE)
  - Web servers (Apache, IIS)
  - Application protocols (HTTP)
- Other network applications have similar components.

Application-Layer Protocols

- Web client software talking to web server software.
  - *Processes* talking to each other.
  - Communicate by sending messages.
- Defines:
  - The types of messages exchanged; requests and responses
  - The syntax of the messages.
  - The meaning of the fields in the message.
  - Rules about how and when processes send and respond to messages.

Front and Back-ends

- In normal client/server models, we always have:
  - the *front-end* on the client side - so called because it is the interface front to the real processing.
  - the *back-end* on the server side - so called because in most cases it is where the required processing is done in the background.

Identifying Resources on the Web (the URL)

- Resources on the web, most commonly web pages, are identified using Uniform Resource Locators (URLs)

- URLs are similar to filenames, except they include:
    - the name of the server they are on
    - network protocols used to access them

Components of a URL

- A URL has the following general form:

  <scheme> : <scheme-specific name>

- Most schemes have the following format:

  <protocol>://<user>:<password>@<host>:<port>/<path>

- When using common browsers, we usually only specify the host and path.
  - The protocol is assumed to be *HTTP*, and the port number is 80.
  - The browser adds on this assumed information to the requests.

Example URLs

http://www.w3.org/hypertext/WWW/TheProject.html

http://www.it.murdoch.edu.au:8888/~jsmith/index.html

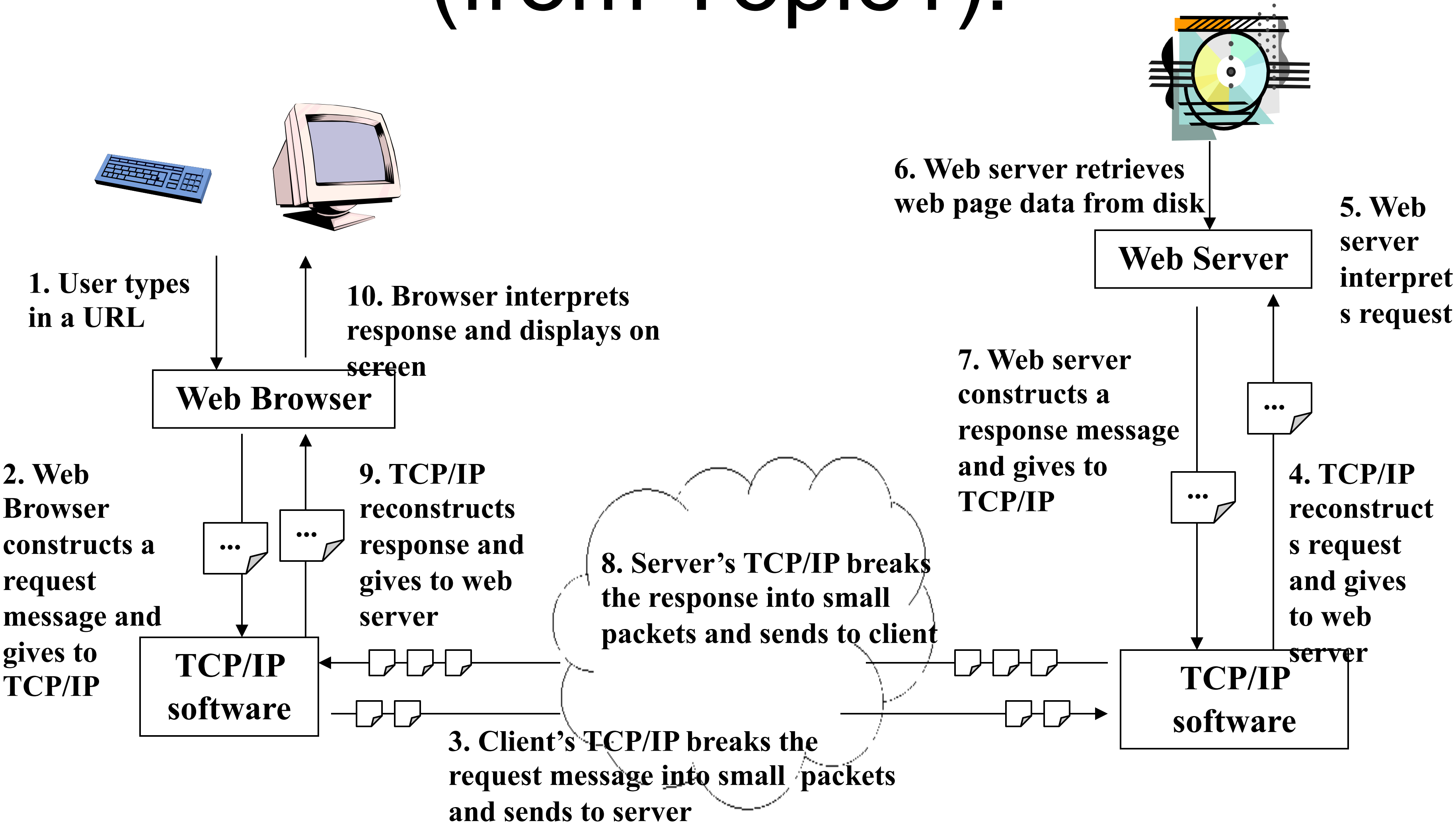
ftp://jsmith@ftp.it.murdoch.edu.au/pub/paper.txt

mailto:J.Smith@murdoch.edu.au

Some Common WWW Schemes

- http
  - access any web resources (html pages, text files, programs, etc.)
- ftp
  - get or put a permanent copy of the required resource
- mailto
  - send an email message to the specified URL.
- file
  - access any file on the local file system (on the same machine/operating system as the web

Web Page Access (from Topic1):



1. User types in a URL

Web Browser

2. Web Browser constructs a request message and gives to TCP/IP

TCP/IP software

3. Client's TCP/IP breaks the request message into small packets and sends to server

TCP/IP software

4. TCP/IP reconstructs request and gives to web server

Web Server

5. Web server interprets request

6. Web server retrieves web page data from disk

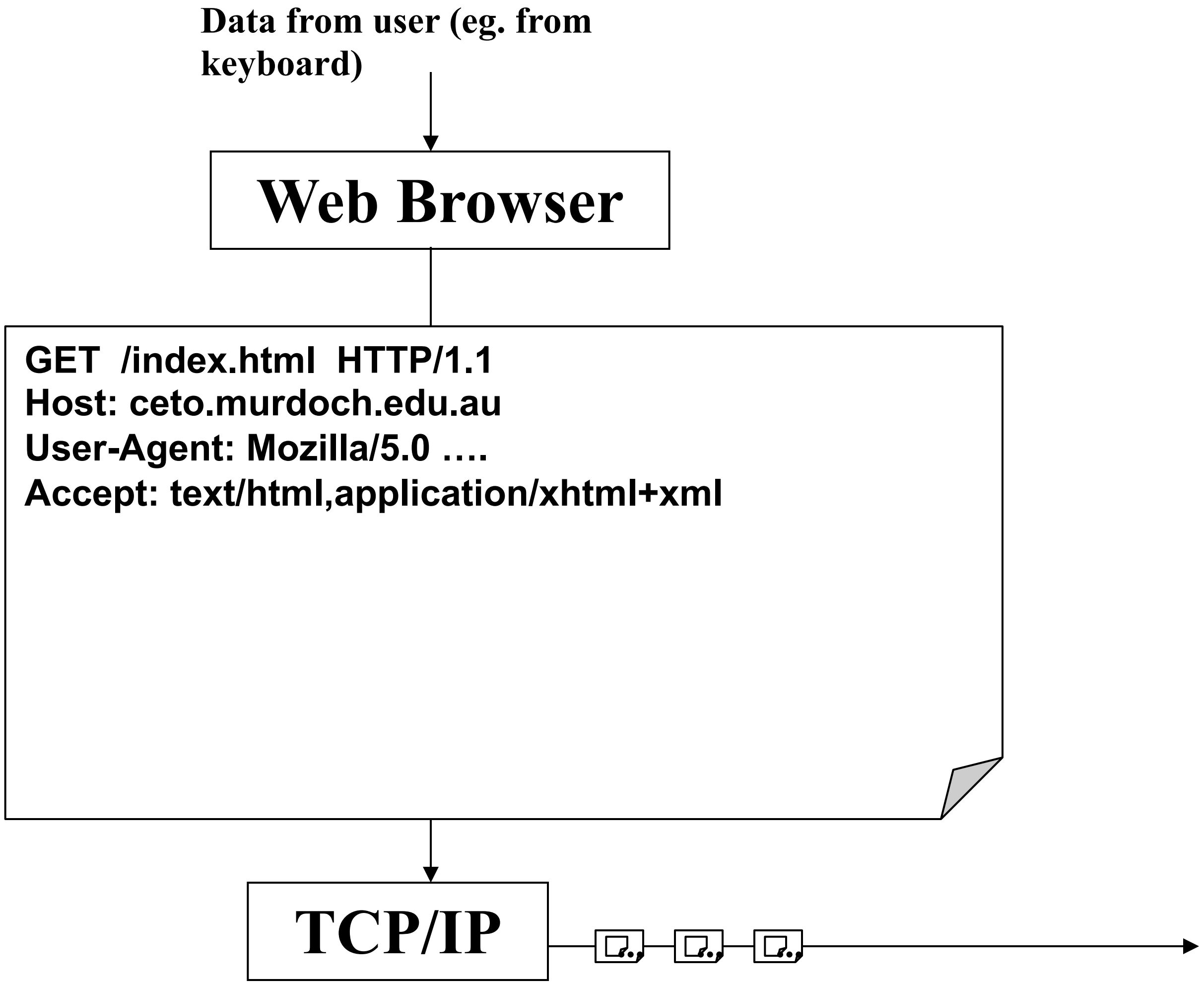
7. Web server constructs a response message and gives to TCP/IP

8. Server's TCP/IP breaks the response into small packets and sends to client

9. TCP/IP reconstructs response and gives to web server

10. Browser interprets response and displays on screen

What happens in Step 2:



Data from user (eg. from keyboard)

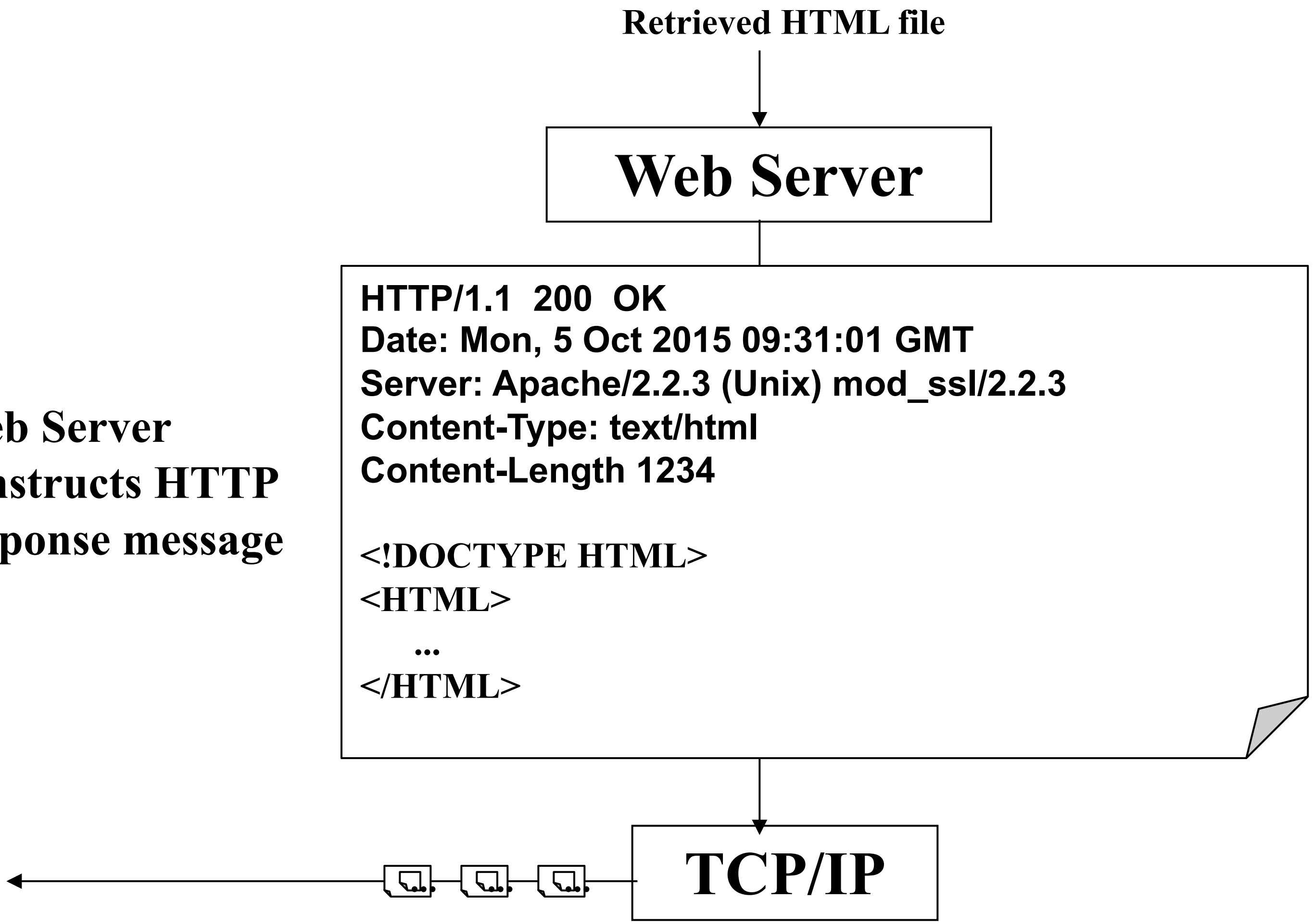
Web Browser

```
GET  /index.html  HTTP/1.1
Host: ceto.murdoch.edu.au
User-Agent: Mozilla/5.0 ….
Accept: text/html,application/xhtml+xml
```

TCP/IP

Web Browser constructs HTTP request message

What happens in Step 7:



Retrieved HTML file

Web Server

```
HTTP/1.1  200  OK
Date: Mon, 5 Oct 2015 09:31:01 GMT
Server: Apache/2.2.3 (Unix) mod_ssl/2.2.3
Content-Type: text/html
Content-Length 1234

<!DOCTYPE HTML>
<HTML>
     ...
</HTML>
```

TCP/IP

Web Server constructs HTTP response message

The Web Server

A Web server is basically an HTTP server

- listens on port 80
- to serve requests made by clients (usually browsers) – eg. for access to HTML pages.
- the format of the data exchanged is based on MIME (Multipurpose Internet Mail Extensions) - a specification for formatting non-ASCII messages so that they can be sent over the Internet. (ref: Webopedia)

The Hypertext Transfer Protocol (HTTP)

- The default protocol for transferring data on the web is HTTP
- In an HTTP interaction, we have one request, and one response.
- Each request-response transaction is independent of each other
  - The server/client must use other ways of keeping track of state information if they need it (eg. using cookies).
  - Note: the transactions at the request-response level is independent, but these days <u>not</u> at the TCP connection level.

The Hypertext Transfer Protocol (HTTP)

- There are two versions of HTTP: the older HTTP/1.0 and the latest version HTTP/1.1.
  - Most browsers today support both.
- HTTP is jointly developed by W3C and IETF.
  - See http://www.w3.org/Protocols/
- HTTP communication happens in the form of request/response exchanges.
  - The requests and responses are in ASCII text.

What does HTTP define?

At the most basic level, HTTP defines that:

- A web client (user agent) initiates communication by sending a request to the web server.
- The format of the request is precisely defined. All information the client wants the server to know is put in the request message.
- A web server when it receives a request message, determines what to do based on what is in the request message.
- The server responds to the web client by sending a response.
- The format of the response is precisely defined. All information the client asked for, as well as other information the server wants the client to know (eg. error descriptions) is put in the response message.

For more details, refer to the introduction in the HTTP/1.1 specifications:

- http://www.w3.org/Protocols/rfc2616/rfc2616-sec1.html

MIME in HTTP

- HTTP transfers requests and responses in a format based on MIME (Multipurpose Internet Mail Extensions).

- Since MIME is a specifications for email, most HTTP messages look like email messages.

- HTTP messages consists of
  - A request line – ie. what the message is about
  - Message headers - ie. control information required to transfer the data

HTTP Requests

An example:

```
GET /index.html HTTP/1.1
Connection: Keep-Alive
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
image/png, */*
Accept-Charset: iso-8859-1,*,utf-8
Accept-Encoding: gzip
Accept-Language: en
Host: ceto.murdoch.edu.au:12345
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows NT; DigExt)
```

HTTP Request Line

The *request-line* in a HTTP request (the first line in the request) consists of

- a *method* to be executed
- a resource to execute on (very commonly a web page)
- the version of HTTP used by the web browser

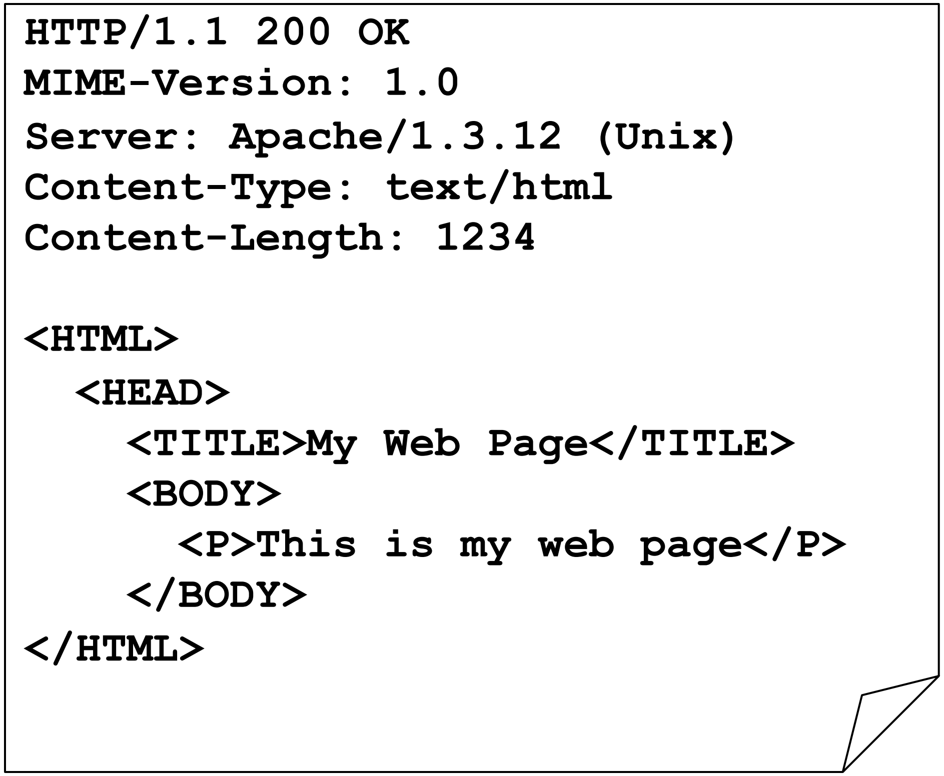
```
GET  /index.html  HTTP/1.1
```

HTTP Methods

- HTTP/1.1 servers and clients must implement the following two methods
    - GET: retrieve a resource from the server and send to client.
    - HEAD: same request as GET, but only want the headers in the response.
- Other optional methods:
    - POST: send data from client to server (eg. when user fills out an HTML form on the client) by appending to a resource.
    - PUT: request a server to replace a resource.
    - DELETE: request a server to remove a resource
    - TRACE: request a loop-back, by having the server

HTTP Responses

An example:



```
HTTP/1.1 200 OK
MIME-Version: 1.0
Server: Apache/1.3.12 (Unix)
Content-Type: text/html
Content-Length: 1234

<HTML>
  <HEAD>
    <TITLE>My Web Page</TITLE>
    <BODY>
      <P>This is my web page</P>
    </BODY>
</HTML>
```

HTTP Response Status Line

- HTTP responses are similar in format to a request, but instead of a Request-line, it has a Status-line to show the status of the response.

- The status line will contain the version of HTTP used by the web server and the result status code, consisting of a number and a text string describing the status.

```
HTTP/1.1 200 OK
```

HTTP Status Codes

Categories of status codes

Category	Code numbers	Description
Informational	100-199	Application specific
Successful	200-299	Request successful
Redirection	300-399	Client needs to do further action
Client Error	400-499	Problems on client-side
Server Error	500-599	Problems on server-side

HTTP Status Codes

Some common response status codes:

Code #	Code String	Description
200	OK	No error, request succeeded
301	Moved Permanently	Requested resource has moved to another URL permanently
400	Bad Request	The server does not understand the request
403	Forbidden	The client is not allowed to access the resource
404	Not Found	The server cannot find the resource

HTTP Header Fields

- Besides the request-line and status-line, HTTP messages also contain other header fields.

- Some are specific to requests, some to responses, and some are not supported by various clients and servers.

Example Header Fields

Each header line is composed of:

header field name: value.

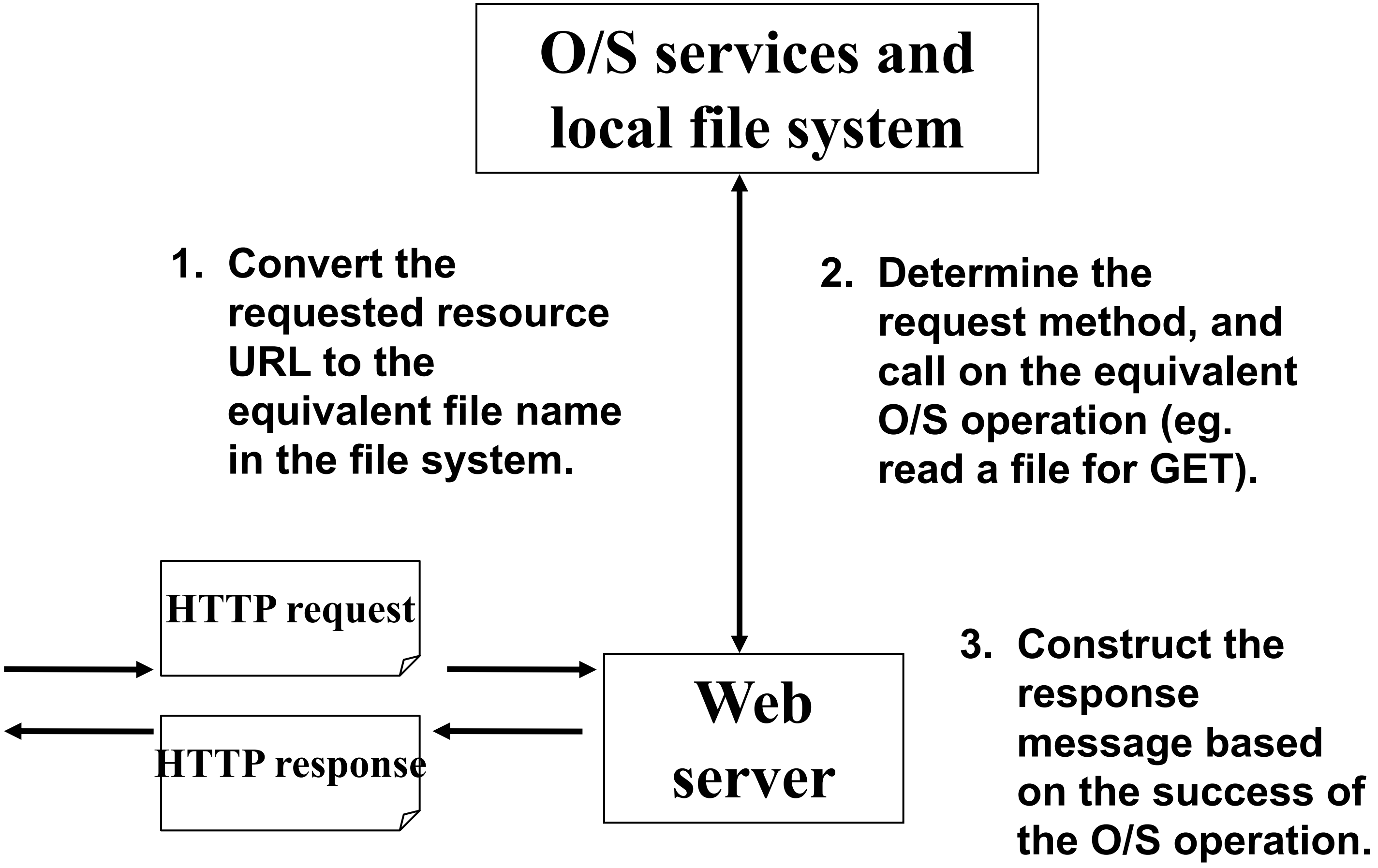
Some example header fields used in most HTTP messages:

- Host: the host that contains the object
- Server: the name and version of the server
- Accept: media types the client is capable of accepting
- Allow: the HTTP methods supported by the server
- Content-Type: the media type of the message body
- Expires: date and time the data in this message becomes invalid

Example Header Fields

- User-Agent: the name and version of the client
- Connection: can indicate that the TCP connection should be closed after the object has been sent (not the default for HTTP1.1)
- Date: the date and time the object was sent
- Content-Length: the length in bytes of the object
- Last-Modified: the date and time when the object was created or last modified
- WWW-Authenticate: defines the authentication needed for the object.
- Accept-language: the preferred language for the data; part of the format negotiation.

Basic Tasks of a Web Server



O/S services and local file system

1. Convert the requested resource URL to the equivalent file name in the file system.
2. Determine the request method, and call on the equivalent O/S operation (eg. read a file for GET).

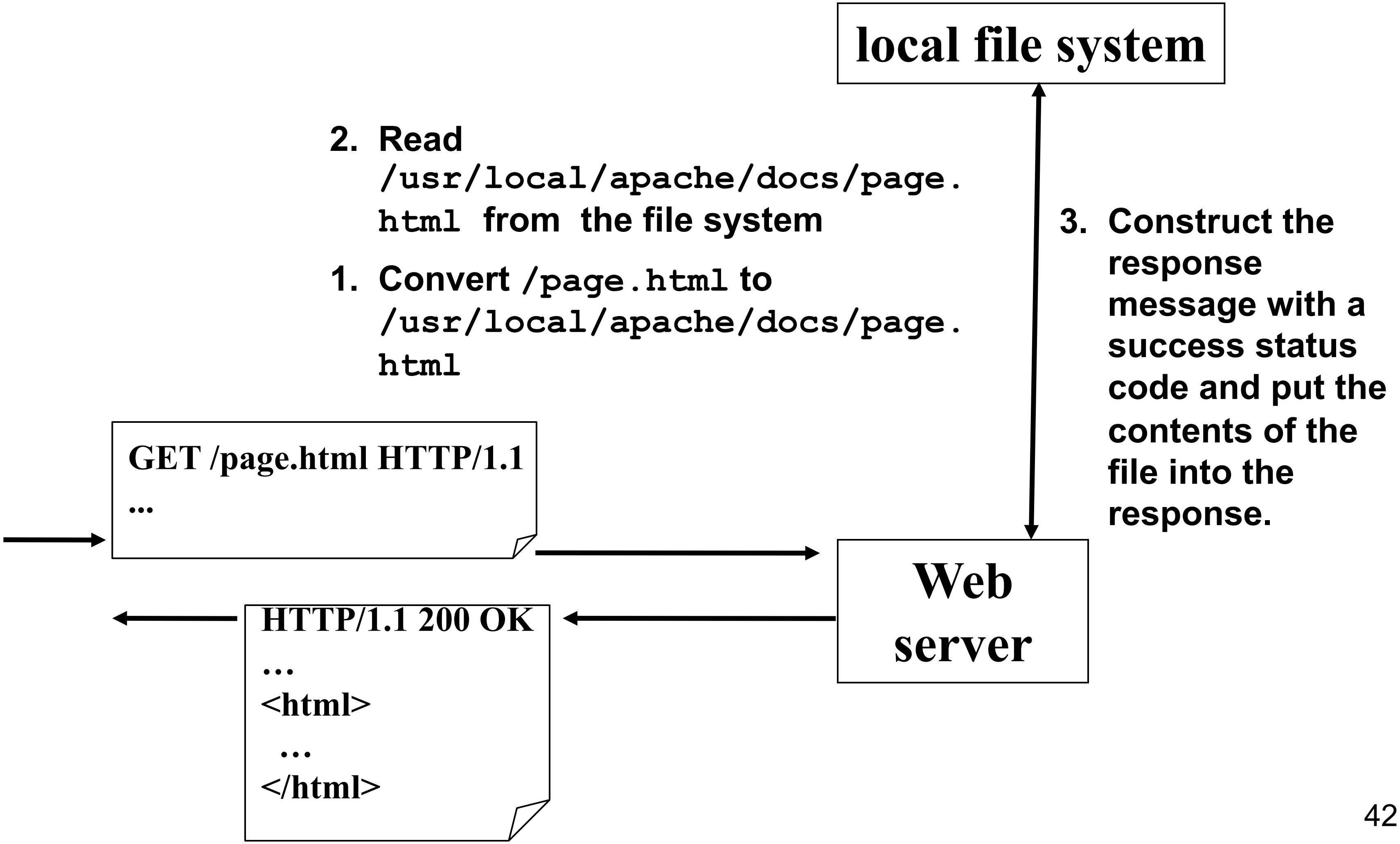
HTTP request

HTTP response

Web server

3. Construct the response message based on the success of the O/S operation.

Example Task of a Web Server



local file system

2. Read `/usr/local/apache/docs/page.html` from the file system

1. Convert `/page.html` to `/usr/local/apache/docs/page.html`

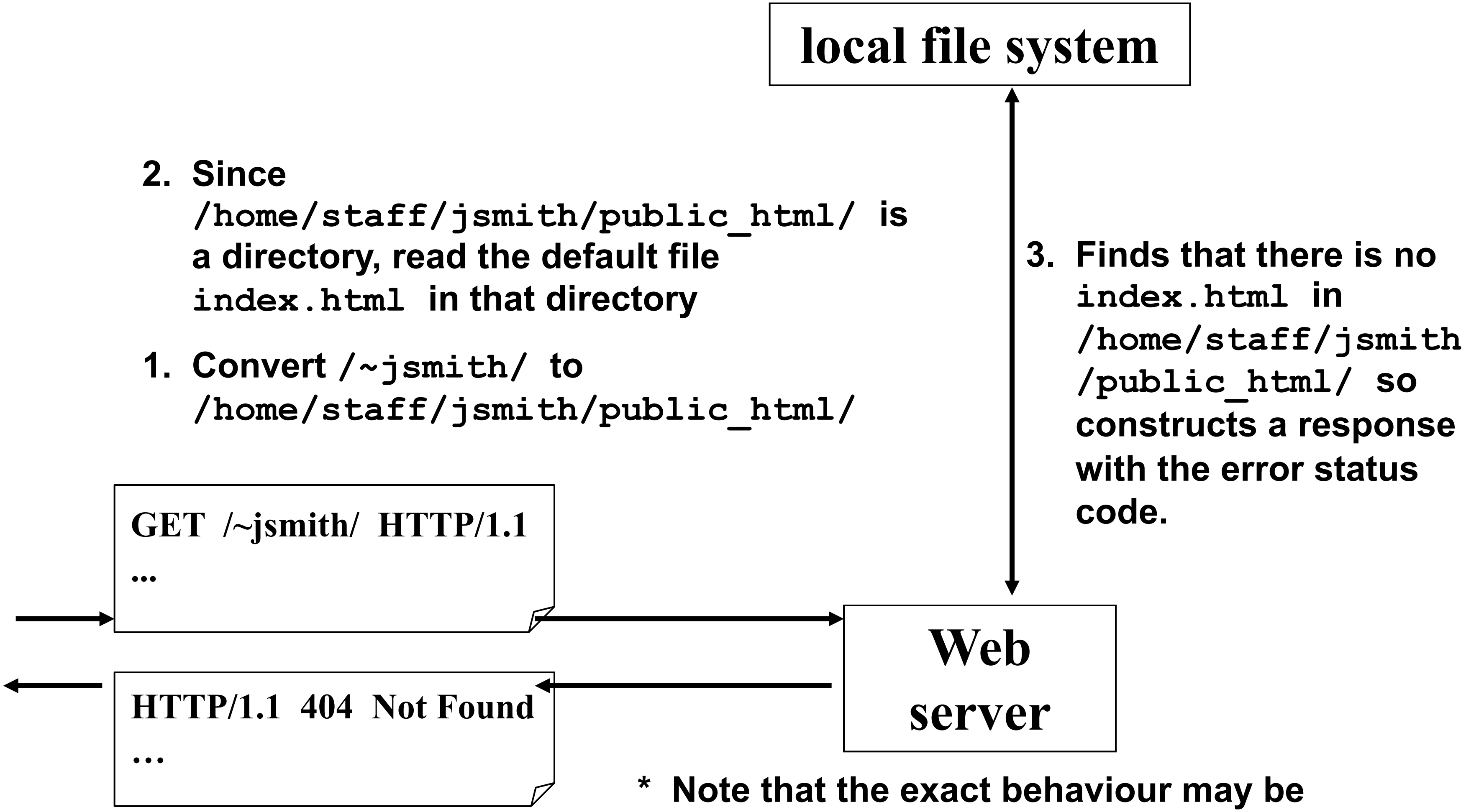
3. Construct the response message with a success status code and put the contents of the file into the response.

```
GET /page.html HTTP/1.1
...
```

Web server

```
HTTP/1.1 200 OK
…
<html>
  …
</html>
```

Example Task of a Web Server



2. Since `/home/staff/jsmith/public_html/` is a directory, read the default file `index.html` in that directory

1. Convert `/~jsmith/` to `/home/staff/jsmith/public_html/`

3. Finds that there is no `index.html` in `/home/staff/jsmith/public_html/` so constructs a response with the error status code.

```
GET /~jsmith/ HTTP/1.1
...
```

```
HTTP/1.1 404 Not Found
…
```

* Note that the exact behaviour may be different depending on how the server is configured.

The Apache HTTP Server

- There are many web servers available, varying greatly in functionality, sophistication, and market penetration.
- The two dominant web servers are Apache HTTP Server from Apache Software Foundation and Internet Information Services from Microsoft.
- Apache is the most popular web server today and it has played a crucial role in the development of the World Wide Web. The majority of web sites in the world are powered by Apache.
- It's free, it's Open Source and it's available in all operating system platforms (including Windows!).

What is HTML?

- HTML stands for HyperText Markup Language. It is the language used to create most of the pages on the web.

- A markup language is used to indicate how the content of the web page should be formatted.

History of HTML

- HTML was defined with SGML
- Original intent of HTML: General layout of documents that could be displayed by a wide variety of computers
  - 1991: Tim Berners-Lee described the first HTML
  - 1993: IETF published first proposal for a HTML specification
  - 1995: IETF published HTML 2.0
  - Jan 1997: HTML 3.2 published as a W3C Recommendation
  - Dec 1997: HTML 4.0 published as a W3C Recommendation
    - Introduced many new features and deprecated many older features
  - Dec 1999: HTML 4.01, which is a cleanup of 4.0, was published as a W3C Recommendation

XHTML and HTML5

- 2000: XHTML 1.0 - Just 4.01 defined using XML, instead of SGML
- 2001: XHTML 1.1 - Modularized 1.0, and drops frames
- 2002: Work on XHTML 2.0 began
- 2004:   Web Hypertext Application Technology Group (WHAT) formed to work on HTML5, a competing standard not based on XHTML
- 2007: Mozilla Foundation, Opera Software and Apple proposed that W3C HTML Working Group adopt WHAT Working Group's HTML5 as the starting point for further HTML development and W3C's next-generation HTML specification be named "HTML 5". W3C accepted the proposal in the following month.
- 2008: W3C released the first HTML5 draft.
- 2009: W3C effectively stopped further development of XHTML 2.0 into a recommendation.
- 2011: the W3C HTML5 Working Draft released.
- 2012: W3C designated HTML5 as a Candidate Recommendation
- October 2014: W3C HTML5 became a W3C recommendation.

Which HTML?

- In the past, we choose to use XHTML 1.0 (or XHTML 1.1 if you do not use frames).
  - XHTML 1.0 and XHTML 1.1 are W3C Recommendations
  - The textbook used XHTML 1.0 and XHTML 1.1

- Starting from 2012, we switch to HTML5.
  - It is now a W3C Recommendation, meaning at there at least two full implementations of the specification.
  - More and more features of HTML5 are implemented by the latest versions of web browsers
  - New web applications will be developed with HTML5 with many new features, such as the support of audios, videos, graphics, location etc..

HTML5 or XHTML5?

- HTML5 has two syntax specifications. One is intended to be backward compatible with the old HTML versions, this is commonly referred to as HTML5.
  - This means any crappy code is acceptable. Even a monkey can write legal HTML5 code!
- The other syntax is based on XML. The resulting HTML5 is called XHTML5.
  - To be legal the HTML5 document must follow strict syntax rules. The documents can potentially be processed by software in areas beyond mere document display.
- Although we have switched to HTML5 in this unit, this is not the license for you to write crappy code.
  - We require you to follow the XHTML5 syntax and every document you code must be validated against XHTML5!

Why XHTML

- HTML has lax syntax rules, leading to sloppy and sometime ambiguous documents
  - XHTML syntax is much more strict, leading to clean and clear documents in a standard form
- HTML processors do not even enforce the few syntax rules that do exist in HTML
  - The syntactic correctness of XHTML documents can be validated
- XHTML encourage the separation of document’s structural description (using XHTML) from document’s presentatioal description (using CSS)

Elements

- Elements are defined by tags (markers)
  - Tag format:
    - Opening tag: <name>
    - Closing tag: </name>
  - The opening tag and its closing tag together specify a container for the *content* they enclose
- Not all tags have content
  - If a tag has no content, its form is <name />
- The container and its content together are called an *element*
- If a tag has attributes, they appear between its name and the right bracket of the opening tag. Eg <p align="center">

Comments

- Comment form: `<!-- … -->`
- Browsers ignore comments, unrecognizable tags, line breaks, multiple spaces, and tabs

HTML Document Structure

- Every XHTML5 document should follow the following structure:

```
<!DOCTYPE html>
<html xmlns = ″http://www.w3.org/1999/xhtml″ >
<head>
   <title> here is the title </title>
</head>
<body>

   Things inside the body is usually rendered.

</body>
</html>
```

Paragraphs

- *Paragraph Elements*
  - The `<p>` tag breaks the current line and inserts a blank line - the new line gets the beginning of the content of the paragraph
  - The browser puts as many words of the paragraph’s content as will fit in each line

```
<!DOCTYPE html>
<!-- greet.hmtl,  a trivial document -->
<html xmlns = ″http://www.w3.org/1999/xhtml″>
  <head>
     <title> Our first document </title>
</head>
  <body>
    <p>
      Greetings from your Webmaster!
    </p>
  </body>
</html>
```

Line Breaks

- Line breaks
  - The effect of the `<br />` tag (no closing tag!) is the same as that of `<p>`, except for the blank line
- Example of paragraphs and line breaks

```
On the plains of hesitation <p> bleach the
bones of countless millions </p> <br />
who, at the dawn of victory <br /> sat down
to wait, and waiting, died.
```

- Typical display of this text:

```
On the plains of hesitation

bleach the bones of countless millions



who, at the dawn of victory
sat down to wait, and waiting, died.
```

Horizontal Rules

- Horizontal rules
  - `<hr />` draws a line across the display, after a line break
  - Example:

```
<hr width="75%" size="20” />
```

- The rule is 75% of the screen width and 20 pixels deep.

Headings

- *Headings*
  - Six sizes, 1 - 6, specified with `<h1>` to `<h6>`
  - 1, 2, and 3 use font sizes that are larger than the default font size
  - 4 uses the default size
  - 5 and 6 use smaller font sizes

```
 <!DOCTYPE html>
<!-- headings.html
     An example to illustrate headings
     -->
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head>
    <title> Headings </title>
</head>
  <body>
    <h1> Aidan’s Airplanes (h1) </h1>
    <h2> The best in used airplanes (h2) </h2>
    <h3> "We’ve got them by the hangarful" (h3)
    </h3>
    <h4> We’re the guys to see for a good used
         airplane (h4) </h4>
    <h5> We offer great prices on great planes
         (h5) </h5>
    <h6> No returns, no guarantees, no refunds,
         all sales are final (h6) </h6>
  </body>
</html>
```

Block Quotes

- To set a block of text off from the normal flow and appearance of text
- Browsers often indent, and sometimes italicize

```
<blockquote>
content of block quotes
</blockquote>
```

Preserving Whitespaces

- If you want a block of text to appear as you type it; for example if you have some special formatting, use the <pre> tag
- Note you still need to use character entity references for special characters <, >, and & within <pre> elements.

```
<pre>This text is showing some code:
BEGIN
   SELECT s_id, s_name
   FROM student
   WHERE s_id = 10045;
END;
</pre>
```

Font Styles and Sizes

- Boldface - `<b>`
- Italics - `<i>`
- Larger - `<big>`
- Smaller - `<small>`
- Monospace - `<tt>`
- Example:

```
The <big> sleet <big> in <big> <i> Crete
</i><br /> lies </big> completely </big>
in </big> the street
```

The sleet in *Crete*

lies completely in the street

Superscripts and subscripts

- Subscripts with `<sub>`
- Superscripts with `<sup>`
- All of these font sizes and font styles can be done with style sheets, but these tags are not yet deprecated

- Example:

```
x<sub>2</sub><sup>3</sup>
```

Display:

x_2^3

Character Entities

Char.	*Entity*	*Meaning*
&	`&amp;`	Ampersand
<	`&lt;`	Less than
>	`&gt;`	Greater than
”	`&quot;`	Double quote
’	`&apos;`	Single quote
¼	`&frac14;`	One quarter
½	`&frac12;`	One half
¾	`&frac34;`	Three quarters
°	`&deg;`	Degree
(space)	`&nbsp;`	Non-breaking space

Images

- GIF (Graphic Interchange Format)
  - 8-bit color (256 different colors)
- JPEG (Joint Photographic Experts Group)
  - 24-bit color (16 million different colors)
- Both use compression, but JPEG compression is better
- Images are inserted into a document with the `<img />` tag with the `src` attribute
  - The `alt` attribute is required by XHTML. The purposes are:
    1. Non-graphical browsers
    2. Browsers with images turned off

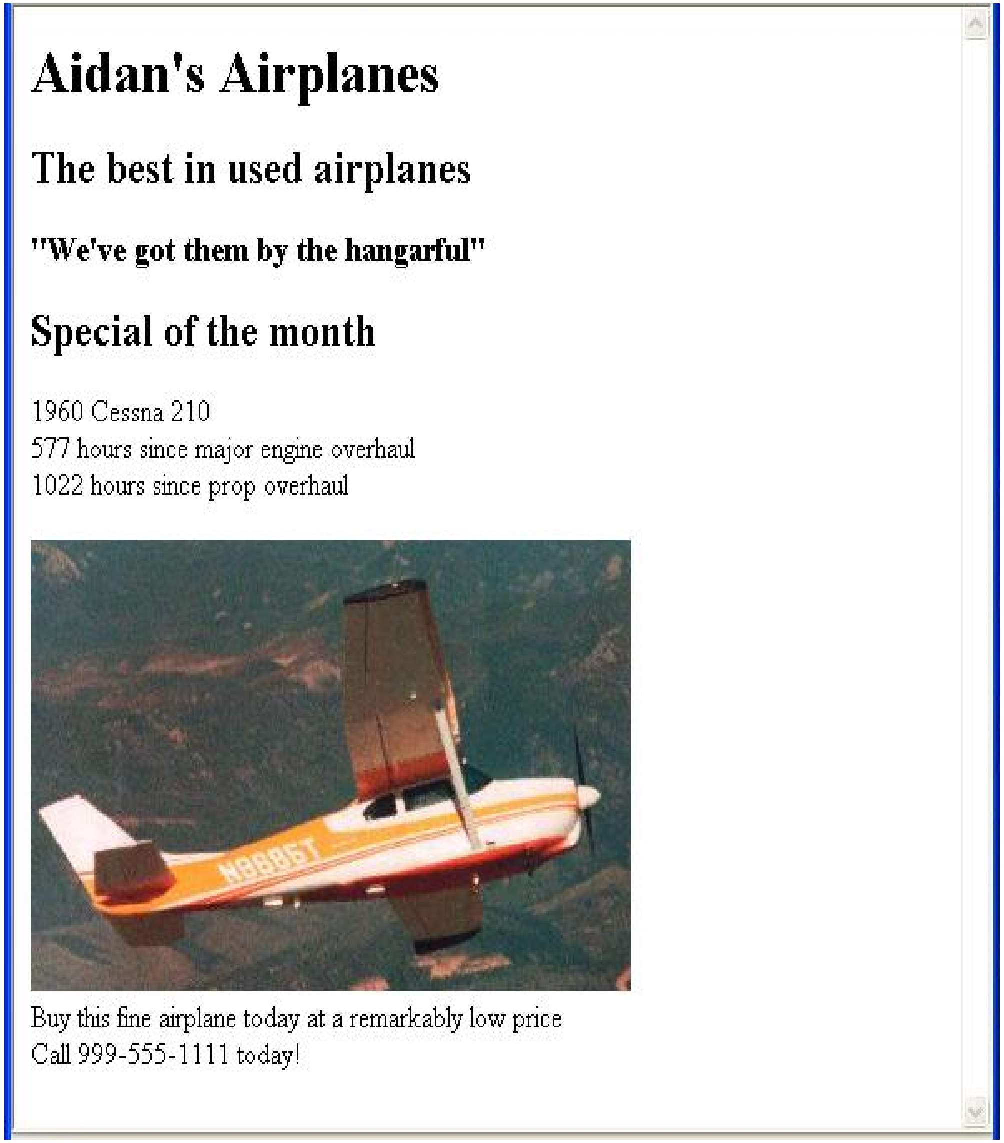
```
<img src = "comets.jpg” alt = "Picture of comets" />
```

- The `<img>` tag has 30 different attributes, including `width` and `height` (in pixels)
- Portable Network Graphics (PNG)
  - Relatively new
  - Should eventually replace both gif and jpeg

Images (cont'd)

```
<!DOCTYPE html>
<!-- image.html, an example to illustrate an image -->
<html xmlns = ”http://www.w3.org/1999/xhtml”>
  <head>
    <title> Images </title>
</head>
  <body>
    <h1> Aidan's Airplanes </h1>
    <h2> The best in used airplanes </h2>
    <h3> "We've got them by the hangarful” </h3>
    <h2> Special of the month </h2>
    <p>
      1960 Cessna 210 <br />
      577 hours since major engine overhaul
      <br />
      1022 hours since prop overhaul
      <br /><br />
      <img src = "c210new.jpg”  alt = "Picture of a Cessna 210"/>
      <br />
      Buy this fine airplane today at a
      remarkably low price <br />
      Call 999-555-1111 today!
    </p>
  </body>
</html>
```

Images (cont'd)



Aidan's Airplanes

The best in used airplanes

"We've got them by the hangarful"

Special of the month

1960 Cessna 210
577 hours since major engine overhaul
1022 hours since prop overhaul

Buy this fine airplane today at a remarkably low price
Call 999-555-1111 today!

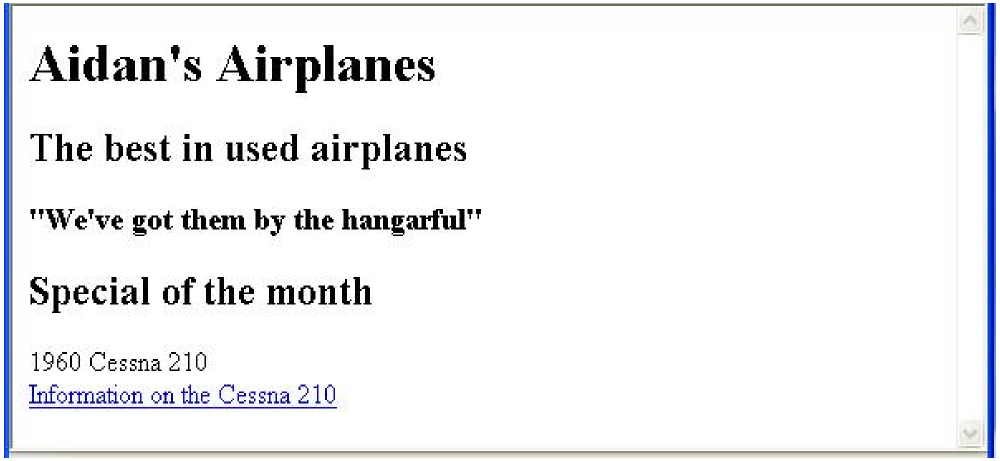
Hypertext Links

- Hypertext is the essence of the Web!
- A link is specified with the `href` (*h*ypertext *ref*erence) attribute of `<a>` (the anchor tag)
- The content of `<a>` is the visual link in the document
- Note: relative addressing of targets is easier to maintain and more portable than absolute addressing

Hypertext Links (cont'd)

```
<!DOCTYPE html>
<!-- link.html, an example to illustrate a link -->
<html xmlns = ″http://www.w3.org/1999/xhtml″>
  <head>
     <title> Links </title>
</head>
  <body>
    <h1> Aidan's Airplanes </h1>
    <h2> The best in used airplanes </h2>
    <h3> "We've got them by the hangarful"
    </h3>
    <h2> Special of the month </h2>
    <p>
      1960 Cessna 210 <br />
      <a href = "C210data.html">
        Information on the Cessna 210 </a>
    </p>
  </body>
</html>
```

Hypertext Links (cont'd)



Aidan's Airplanes

The best in used airplanes

"We've got them by the hangarful"

Special of the month

1960 Cessna 210
Information on the Cessna 210

Targets within Documents

- If the target is not at the beginning of the document, the target spot must be marked
- Target labels can be defined in many different tags with the `id` attribute, as in

```
  <h1 id = "baskets"> Baskets </h1>
```

- The link to an `id` must be preceded by a pound sign (`#`); If the `id` is in the same document, this target could be

```
  <a href = "#baskets"> What about baskets? </a>
```

- If the target is in a different document, the document reference must be included

```
  <a href = "myAd.html#baskets”> Baskets </a>
```

- Style note: a link should blend in with the surrounding text, so reading it without taking the link should not be made less pleasant
- Links can have images:

```
<a href = "c210data.html“>
     <img src = "smallplane.jpg”
             alt = "Small picture of an airplane “ />
     Info on C210 </a>
```

Page Opened in a New Window

- All the links we have considered so far have opened in the same window as the current window.
- Sometimes you want the linked page to open in a separate window (the original will still be open). Do this with the target attribute.
- Example:

```
<a href=“IntroToIct286.index”

       target=“ict286”>  ICT286 </a>
```

Links to Files in Folders

- Links can be made to web pages in other directories. The current directory or folder is the one that contains the HTML file we are currently using.
- We will consider
  - Linking to a file in a child folder – one below the current one.
  - Linking to a file in a folder above the current one (parent)
  - Linking to a file in another folder on the same level as the current one (sibling)
- Please note you can use either absolute path or relative path to reference a local file, but it is usually a bad idea to use an absolute path. This is because if you move the web site to a different folder or to a different machine, the link would stop working.

Links to Files in Folders (cont'd)

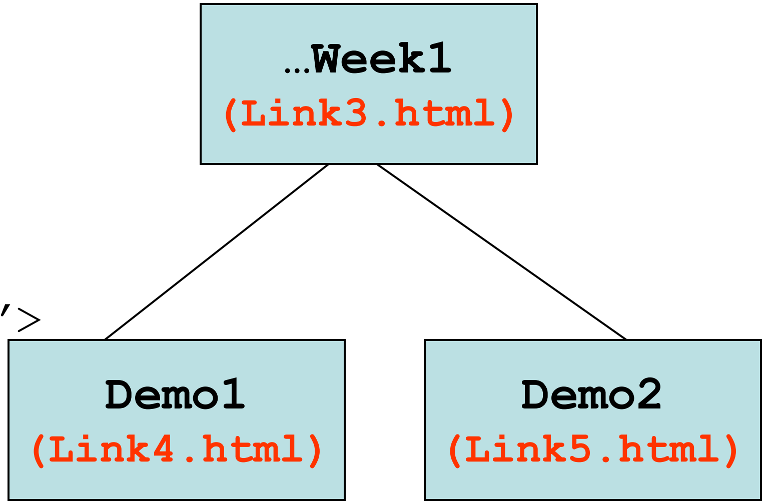
- Consider the directory structure below, that we will use for the next few slides:
  - In file Link3.html:

    -
      ```
      <a href=“Demo1/Link4.html”>
        Demo 1</a>
      ```

    -
      ```
      <a href=“Demo2/Link5.html”>
        Demo 2</a>
      ```
  - In file Link4.html:

    -
      ```
      <a href=“../Link3.html”>
           Week 1</a>
      ```

    -
      ```
      <a href=“../Demo2/Link5.html”>
           Demo 2</a>
      ```



Mailto Links

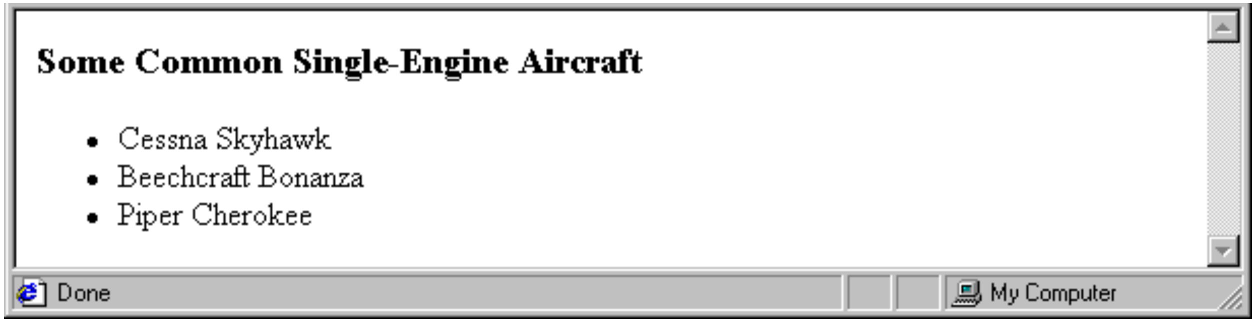
- Links can be made so that someone can easily email you; see example below:

```
<a href="mailto:h.xie@murdoch.edu.au">
   email the lecturer </a>
```

Unordered Lists

- The list is the content of the `<ul>` tag
- List elements are the content of the `<li>` tag

```
<h3> Some Common Single-Engine Aircraft </h3>
 <ul>
   <li> Cessna Skyhawk </li>
   <li> Beechcraft Bonanza </li>
   <li> Piper Cherokee </li>
 </ul>
```



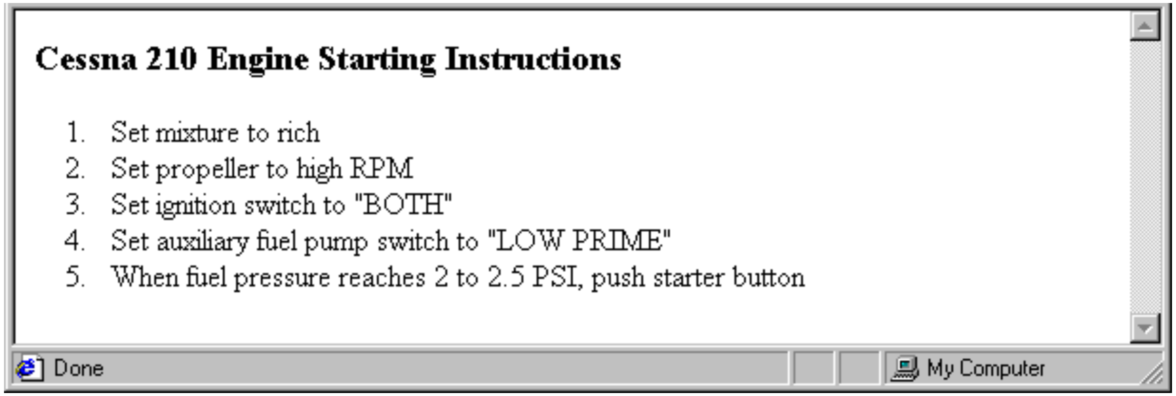
Some Common Single-Engine Aircraft

- Cessna Skyhawk
- Beechcraft Bonanza
- Piper Cherokee

Ordered Lists

- The list is the content of the `<ol>` tag
- Each item in the display is preceded by a sequence value

```
<h3> Cessna 210 Engine Starting Instructions
</h3>
<ol>
  <li> Set mixture to rich </li>
  <li> Set propeller to high RPM </li>
  <li> Set ignition switch to "BOTH" </li>
  <li> Set auxiliary fuel pump switch to
       "LOW PRIME" </li>
  <li> When fuel pressure reaches 2 to 2.5
       PSI, push starter button </li>
</ol>
```



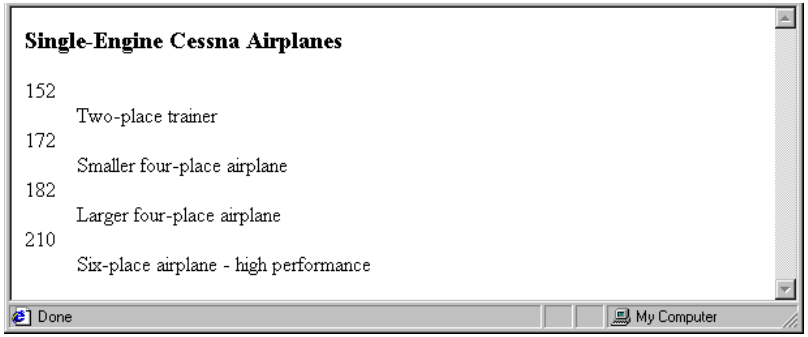
Cessna 210 Engine Starting Instructions

1. Set mixture to rich
2. Set propeller to high RPM
3. Set ignition switch to "BOTH"
4. Set auxiliary fuel pump switch to "LOW PRIME"
5. When fuel pressure reaches 2 to 2.5 PSI, push starter button

Definition Lists

- List is the content of the `<dl>` tag
- Terms being defined are the content of the `<dt>` tag
- The definitions themselves are the content of the `<dd>` tag

```
<h3> Single-Engine Cessna Airplanes </h3>
<dl >
  <dt> 152 </dt>
  <dd> Two-place trainer </dd>
  <dt> 172 </dt>
  <dd> Smaller four-place airplane </dd>
  <dt> 182 </dt>
  <dd> Larger four-place airplane </dd>
  <dt> 210 </dt>
  <dd> Six-place airplane - high performance
  </dd>
</dl>
```



Single-Engine Cessna Airplanes

152
    Two-place trainer
172
    Smaller four-place airplane
182
    Larger four-place airplane
210
    Six-place airplane - high performance

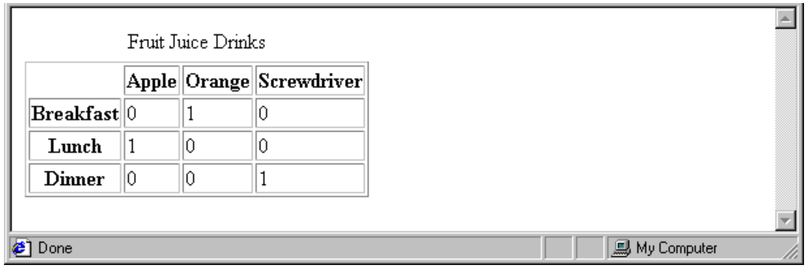
Tables

- A table is a matrix of cells, each possibly having content
- The cells can include almost any element
- Some cells have row or column labels and some have data
- A table is specified as the content of a `<table>` tag
- A `border` attribute in the `<table>` tag specifies a border between the cells
- If `border` is set to `"border"`, the browser’ s default width border is used
- The `border` attribute can be set to a number, which will be the border width
- Without the `border` attribute, the table will have no lines!
- Tables are given titles with the `<caption>` tag, which can immediately follow `<table>`

Tables (cont'd)

- Each row of a table is specified as the content of a `<tr>` tag
- The row headings are specified as the content of a `<th>` tag
- The contents of a data cell is specified as the content of a `<td>` tag

```
<table border = "border">
   <caption> Fruit Juice Drinks </caption>
     <tr>
       <th> </th>
       <th> Apple </th>
       <th> Orange </th>
       <th> Screwdriver </th>
     </tr>
     <tr>
       <th> Breakfast </th>
       <td> 0 </td>
       <td> 1 </td>
       <td> 0 </td>
     </tr>
     <tr>
       <th> Lunch </th>
       <td> 1 </td>
       <td> 0 </td>
       <td> 0 </td>
     </tr>
      . . .
   </table>
```



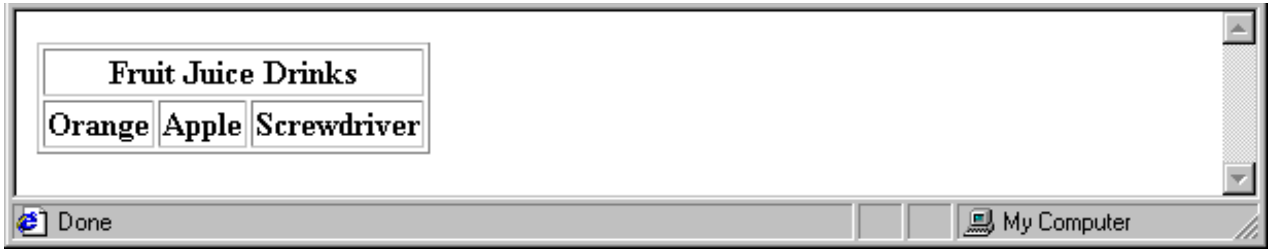
Fruit Juice Drinks

	Apple	Orange	Screwdriver
Breakfast	0	1	0
Lunch	1	0	0
Dinner	0	0	1

Tables (cont'd)

- A table can have two levels of column labels
  - If so, the `colspan` attribute must be set in the `<th>` tag to specify that the label must span some number of columns

```
<tr>
  <th colspan = "3"> Fruit Juice Drinks </th>
</tr>
<tr>
  <th> Orange </th>
  <th> Apple </th>
  <th> Screwdriver </th>
</tr>
```

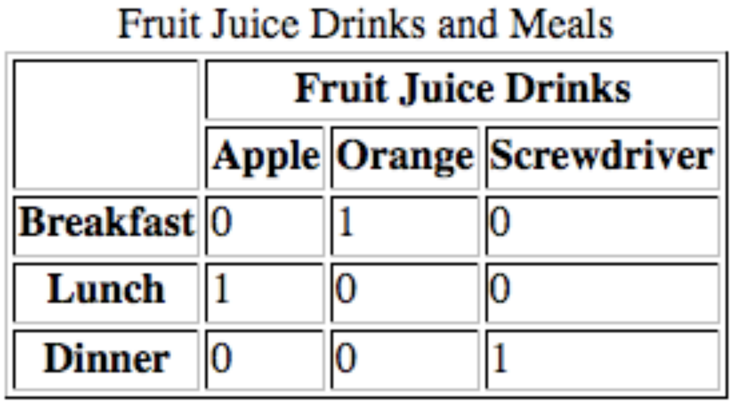


Fruit Juice Drinks		
Orange	Apple	Screwdriver

Tables (cont'd)

- If the rows have labels and there is a spanning column label, the upper left corner must be made larger, using `rowspan`

```
<table border = "border">
  <tr>
    <td rowspan = "2"> </td>
    <th colspan = "3"> Fruit Juice Drinks
    </th>
  </tr>
  <tr>
    <th> Apple </th>
    <th> Orange </th>
    <th> Screwdriver </th>
  </tr>
    …
</table>
```



Fruit Juice Drinks and Meals

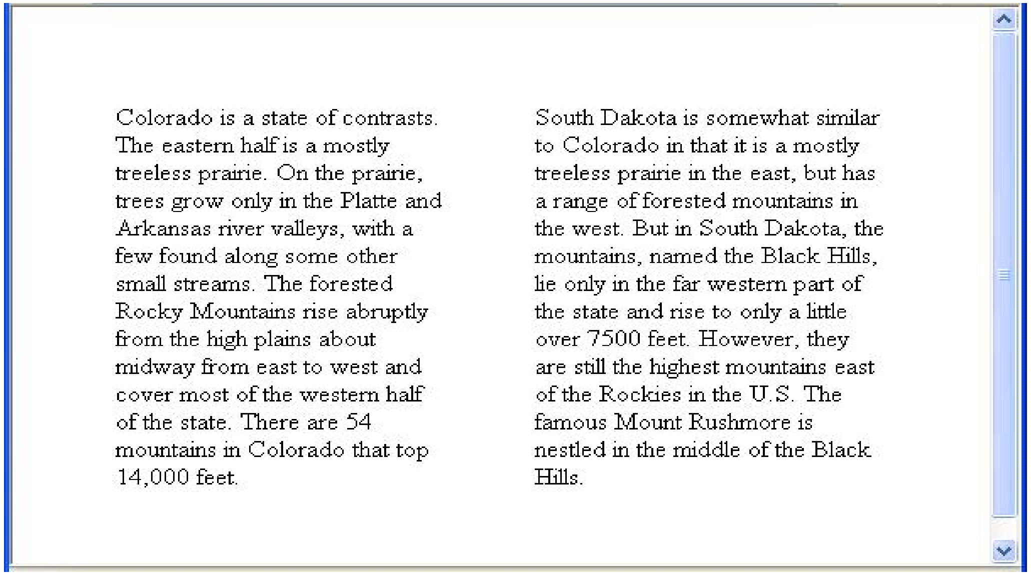
	Fruit Juice Drinks		
	Apple	**Orange**	**Screwdriver**
Breakfast	0	1	0
Lunch	1	0	0
Dinner	0	0	1

Tables (cont'd)

- The `align` attribute controls the horizontal placement of the contents in a table cell
  - Values are `left`, `right`, and `center` (default)
  - `align` is an attribute of `<tr>`, `<th>`, and `<td>` elements
- The `valign` attribute controls the vertical placement of the contents of a table cell
  - Values are `top`, `bottom`, and `center` (default)
  - `valign` is an attribute of `<th>` and `<td>` elements
- The `cellspacing` attribute of `<table>` is used to specify the distance between cells in a table
- The `cellpadding` attribute of `<table>` is used to specify the spacing between the content of a cell and the inner walls of the cell

Tables (cont'd)

```
<table cellspacing = "50">
     <tr>
        <td> Colorado is a state of …
        </td>
        <td> South Dakota is somewhat…
        </td>
     </tr>
  </table>
```



Colorado is a state of contrasts. The eastern half is a mostly treeless prairie. On the prairie, trees grow only in the Platte and Arkansas river valleys, with a few found along some other small streams. The forested Rocky Mountains rise abruptly from the high plains about midway from east to west and cover most of the western half of the state. There are 54 mountains in Colorado that top 14,000 feet.	South Dakota is somewhat similar to Colorado in that it is a mostly treeless prairie in the east, but has a range of forested mountains in the west. But in South Dakota, the mountains, named the Black Hills, lie only in the far western part of the state and rise to only a little over 7500 feet. However, they are still the highest mountains east of the Rockies in the U.S. The famous Mount Rushmore is nestled in the middle of the Black Hills.

Document Validator

- W3C has provided a document validation:http://validator.w3.org/#validate_by_upload
  - It supports the validation for XHTML 1.0 and 1.1. Unfortunately, its support for HTML5 validation is still limited.
- The following website: http://totalvalidator.com provides an excellent and free validator program (Total Validator Basic) that can validate HTML5, XHTML5 etc.
  - You can download it to run on your own machine (all platforms are supported)
- In this unit, you are required to create HTML5 documents manually with a text editor and have it validated against XHTML5 or XHTML5.1. You are not permitted to use WYSIWYG HTML/XML editor (such as Adobe DreamWeaver or Microsoft Expression Web) to create HTML5 documents.

Summary

- The Internet and WWW work together to give users access to a myriad of information and resources.

- Most of the web pages on the WWW are created using HTML.

- An HTML page consists of the content and various tags that are used to format the content

- Use HTML5, but always validate your documents against XHTML5.

Readings

- Sebesta: Chapter 1 and 2
- HTTP Protocol:
  - Official HTTP 1.1 specifications (http://www.w3.org/Protocols/rfc2616/rfc2616.html)
- The Apache HTTP Server:
  - http://httpd.apache.org/
  - Apache 2.4 Documentation (http://httpd.apache.org/docs/2.4/)
- Useful website for HTML5:
  - http://www.w3schools.com/html/html5_intro.asp
- Useful book:
  - Mark Myers: A smart way to learn HTML & CSS (amazon kindle book, costs around $10 from amazon.com.au)