Client-Side Processing

HTML, CSS, Forms, JavaScript, DOM,
Java Applets, ActiveX controls, and AJAX

HTML

HTML

- Specification controlled by W3C:
  - www.w3.org/html
- HTML element list:
  - http://www.w3.org/html/wiki/Elements
- Versions:
  - HTML 4.01 (http://www.w3.org/TR/html401/)
  - XHTML (http://www.w3.org/TR/xhtml1/)
  - HTML 5 (http://dev.w3.org/html5)

HTML 5

- Still a draft specification
- Jointly developed by:
  - W3C
  - WHATWG (Apple, Mozilla and Opera, et.al.)
- Not yet supported in all Browsers
  - try http://html5test.com/
- Backward compatible; supports HTML and XHTML syntax

Changes in HTML5

- New structure elements :
  - `section, article, header, footer, figure, nav, ...`
- New content elements :
  - `canvas, video, audio, embed, mark, progress, meter, time, command, details, datalist, output, ...`
- New input element types :
  - `tel, url, email, date, month, range, color, ...`
- Removed elements :
  - `big, center, font, frame, frameset, applet, ...`
- Other new features:
  - Microdata, Drag-and-drop, ...

HTML Document Types

- HTML 4.01:

```
<! DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<! DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<! DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
```

- XHTML:

```
<?xml version="1.0" encoding="UTF-8"?>
<! DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<! DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<! DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
 <! DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

- HTML 5:

```
<! DOCTYPE html>
```

- http://validator.w3.org/
- Quirks vs Standards modes (see Wikipedia Quicks mode)

CSS

HTML and CSS

- Designed to separate:
  a) Context (HTML) from
  b) Presentation (CSS)

Cascading Style Sheets (CSS)

- A style sheet is made up of style rules that tell a browser how to present a HTML (or XML) document.

```
<HEAD>
    <TITLE>CSS Example</TITLE>
    <STYLE TYPE="text/css">
        H1 { font-size: x-large; color: red }
        H2 { font-size: large; color: blue  }
    </STYLE>
</HEAD>
```

- In CSS, more than one style sheet can influence the presentation simultaneously. There are two main reasons for this feature: modularity and author/reader balance.
- When multiple style sheets are used, the style sheets may fight over control of a particular selector. In these situations, there must be rules as to which style sheet's rule will win out.

How to Insert a Style Sheet

- Including external style sheets:

```
<head>
<link rel="stylesheet" type="text/css" href="MyStyle.css" />
</head>
...
```

MyStyle.css:

```
hr {color: sienna}
p {margin-left: 20px}
body {background-image: url("images/back40.gif")}
```

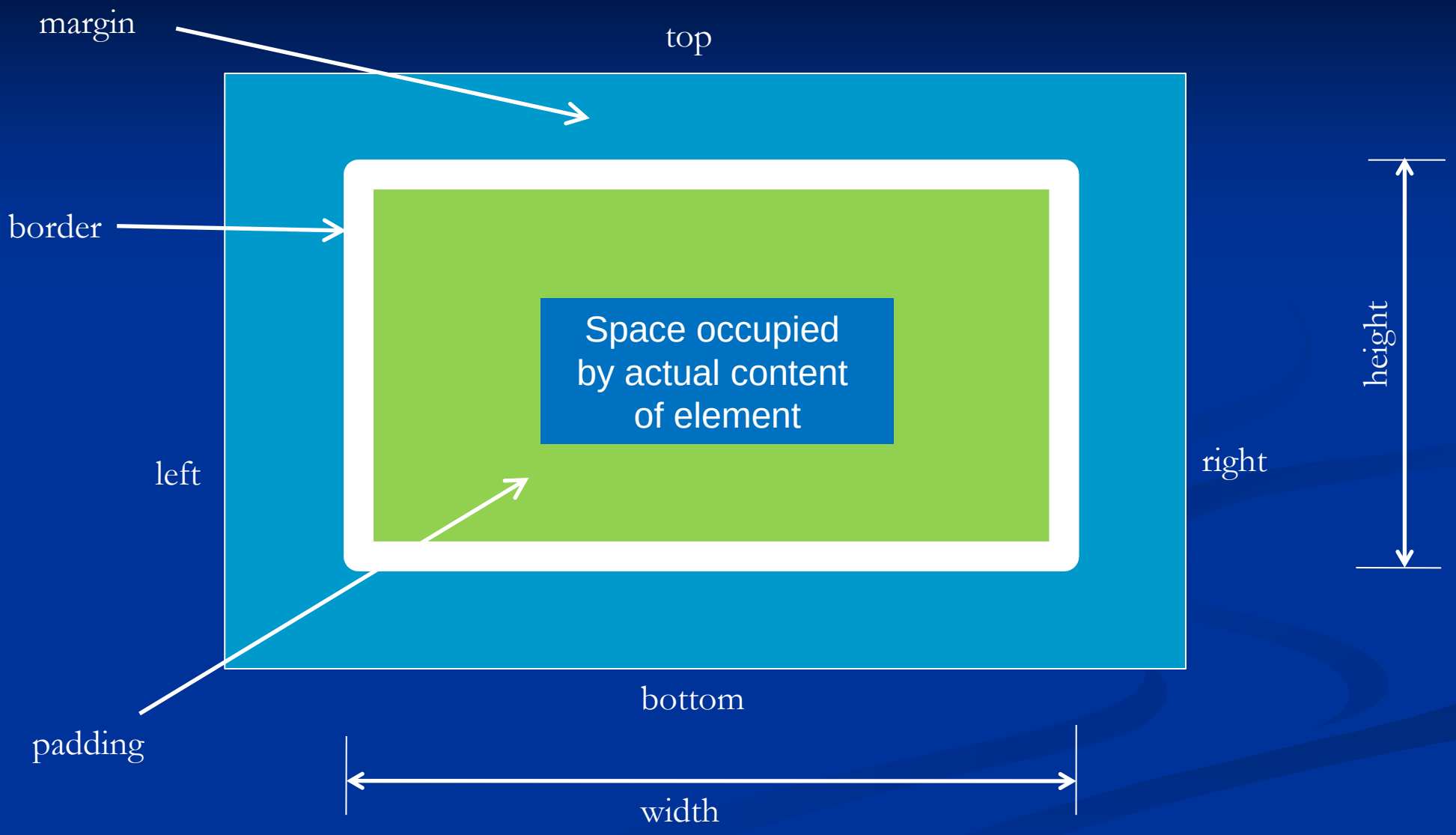
- Inline Styles:

```
<p style="color: #00ff00; margin-left: 20px">
    This is a paragraph
</p>
...
```

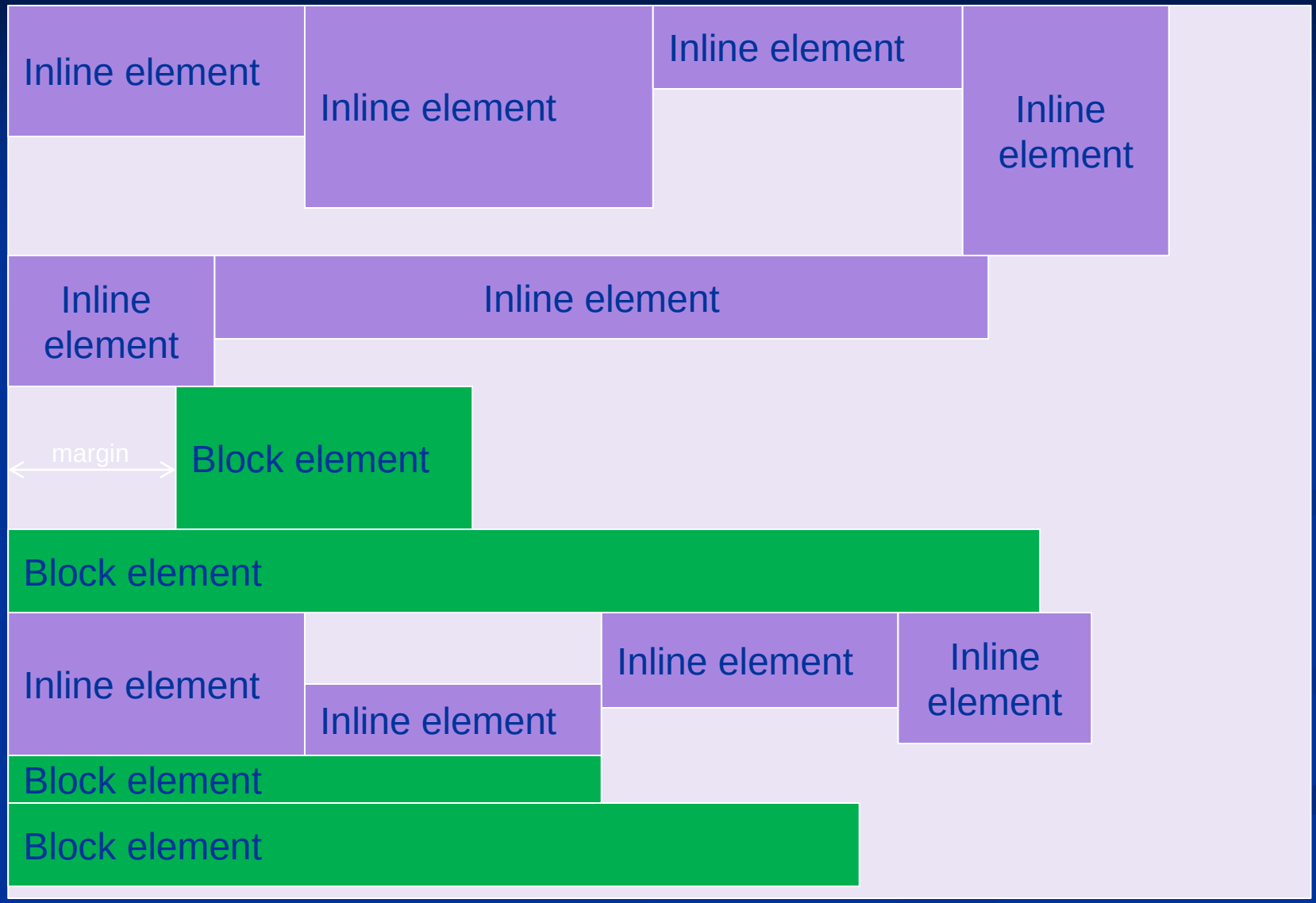
Sample Default Style Sheet for HTML

```
BODY { margin: 1em; font-family: serif; line-height: 1.1; background: white; color: black; }
H1, H2, H3, H4, H5, H6, P, UL, OL, DIR, MENU, DIV, DT, DD, ADDRESS, BLOCKQUOTE, PRE, BR,
HR, FORM, DL        { display: block }
B, STRONG, I, EM, CITE, VAR, TT, CODE, KBD, SAMP, IMG, SPAN { display: inline }
LI        { display: list-item }
H1, H2, H3, H4      { margin-top: 1em; margin-bottom: 1em }
H5, H6    { margin-top: 1em }
H1        { text-align: center }
H1, H2, H4, H6      { font-weight: bold }
H3, H5    { font-style: italic }
H1        { font-size: xx-large }
H2        { font-size: x-large }
H3        { font-size: large }
B, STRONG { font-weight: bolder }
I, CITE, EM, VAR, ADDRESS, BLOCKQUOTE { font-style: italic }
PRE, TT, CODE, KBD, SAMP { font-family: monospace }
PRE       { white-space: pre }
ADDRESS   { margin-left: 3em }
BLOCKQUOTE{ margin-left: 3em; margin-right: 3em }
UL, DIR   { list-style: disc }
OL        { list-style: decimal }
MENU      { margin: 0 } /* tight formatting */
LI        { margin-left: 3em }
DT        { margin-bottom: 0 }
DD        { margin-top: 0; margin-left: 3em }
HR        { border-top: solid }
A:link    { color: blue }
A:visited { color: red }
A:active  { color: lime }
A:link IMG          { border: 2px solid blue }
A:visited IMG       { border: 2px solid red }
A:active IMG        { border: 2px solid lime }
```

The Box Model



Normal Flow


Inline element
Inline element
Inline element
Inline element
Inline element
Inline element
margin
Block element
Block element
Inline element
Inline element
Inline element
Inline element
Block element
Block element

Div and Span

```
H1, H2, H3, H4, H5, H6, P, UL, OL, DIR, MENU, DIV, DT, DD, ADDRESS, BLOCKQUOTE, PRE, BR,
HR, FORM, DL        { display: block }
B, STRONG, I, EM, CITE, VAR, TT, CODE, KBD, SAMP, IMG, SPAN { display: inline }
```

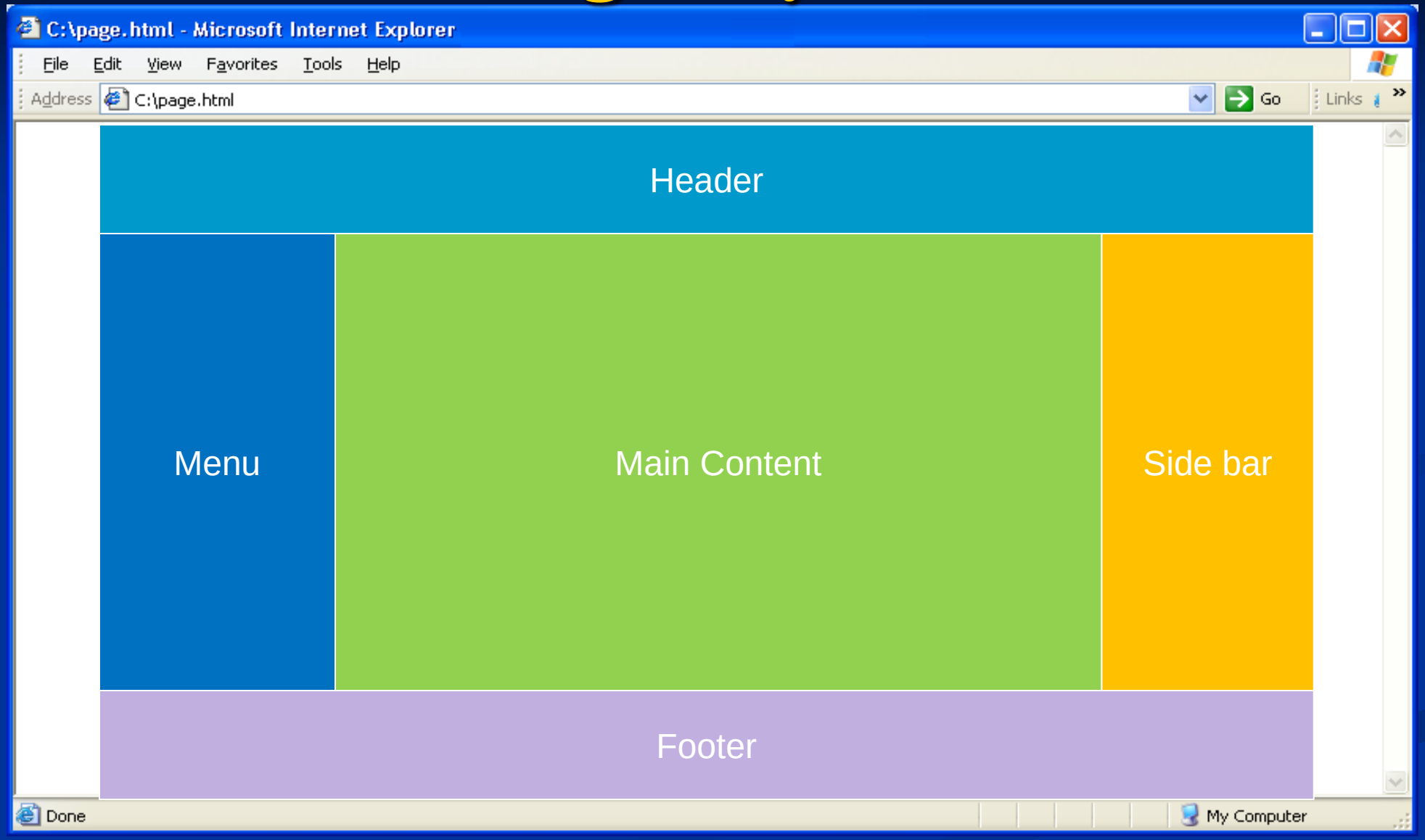
Generic containers:

```
<span>
    nested content
</span>

<div>
    nested content
</div>
```

CSS class vs. id

```
<p id="news" class="important" >
    nested content
</p>
<div class="important">
    nested content
</div>

<style type="text/css">
  p                { color: blue          }
  #news            { bg-color: red        }
 .important        { font-weight: bold    }
  div.important    { margin-left: 10px    }
</style >
```

Page Layout

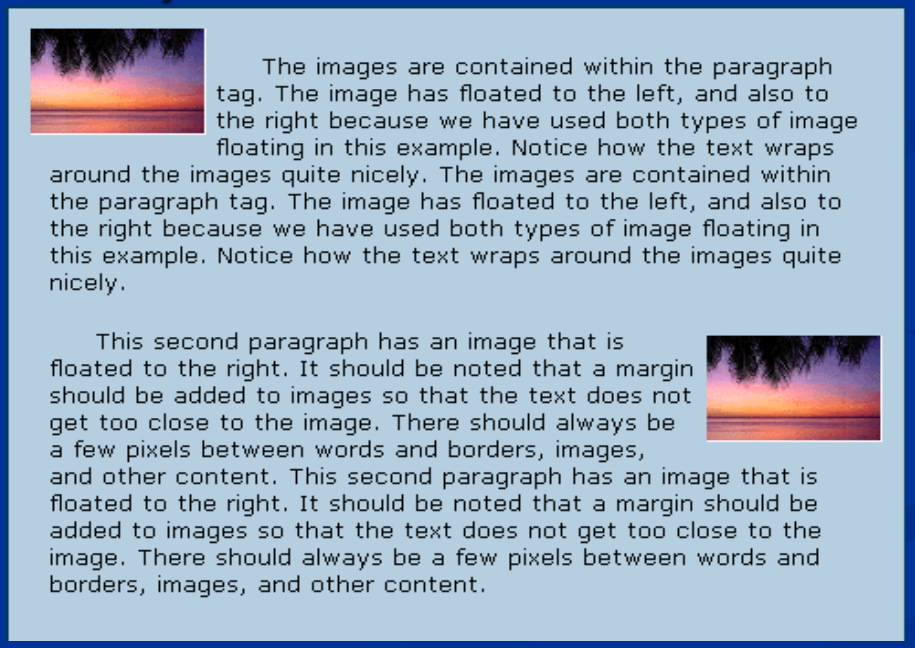

C:\page.html - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\page.html
Go
Links
Header
Menu
Main Content
Side bar
Footer
Done
My Computer

Page layout using DIV and CSS

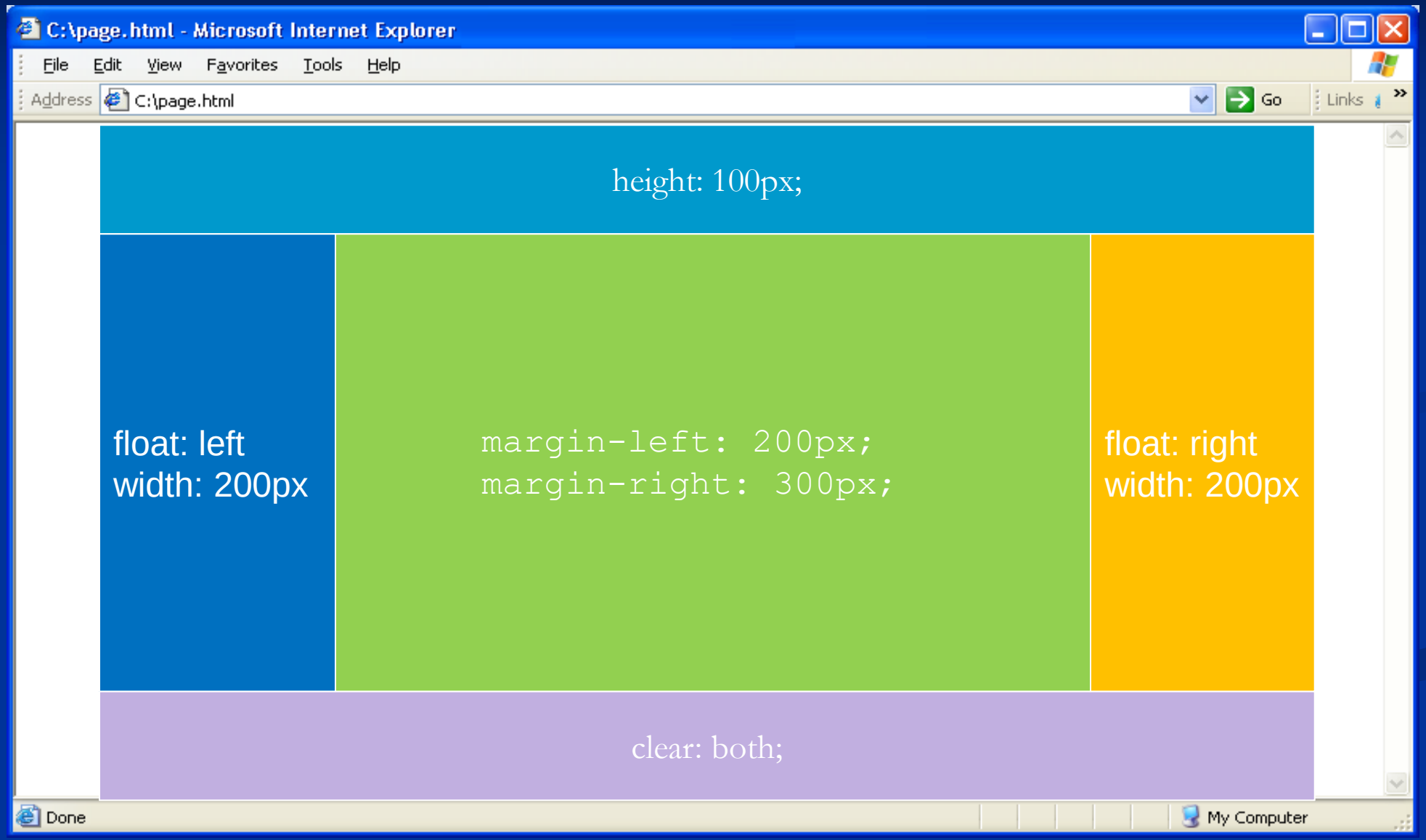
```
<body>
  <div class="header">
     everything that needs to be displayed in the header
  </div>
  <div class="menu">
     html elements to create a menu
  </div>
  <div class="sidebar">
     advertising perhaps?
  </div>
  <div class="content">
     the main content of the page
  </div>
  <div class="footer">
     generic footer info (e.g. last updated, contact us, etc)
  </div>
</body>
```

Floating

- Instead of positioning a box according to normal flow layout, a box can be floated to the left or right.
  - CSS **`float`** property can be **`left`**, **`right`** or **`none`** (the default).
  - The box is moved to the far left or right side of the containing box and other content *flows* around it.
- The following shows an example of two image elements, floated to the left and right respectively:



positioning using CSS



CSS Layout Tutorials

1. http://css-discuss.incutio.com/wiki/Three_Column_Layouts
2. http://matthewjamestaylor.com/blog/perfect-3-column.htm
3. http://www.neuroticweb.com/recursos/3-columns-layout/
4. http://www.maxdesign.com.au/articles/css-layouts/

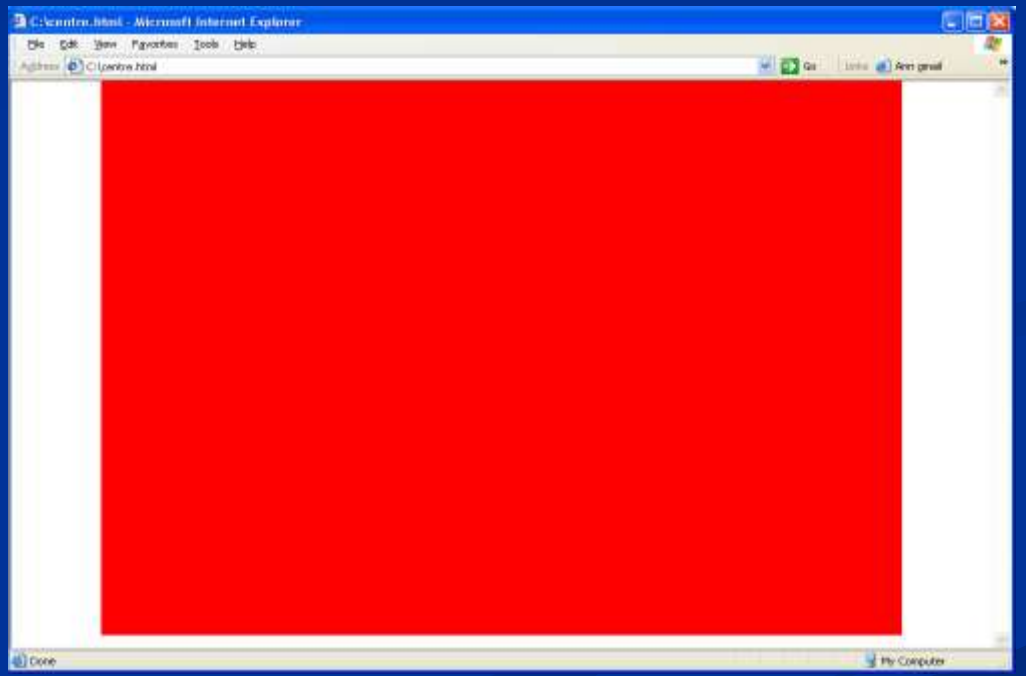
- CSS vs Tables: 13 reasons why CSS is superior to tables
  - http://www.chromaticsites.com/blog/13-reasons-why-css-is-superior-to-tables-in-website-design/
- Why CSS should not be used for layout
  - http://www.flownet.com/ron/css-rant.html

- **CSS Multi-column Layout Module (W3C Candidate Recommendation)**
  - http://www.w3.org/TR/css3-multicol/

Auto Margins

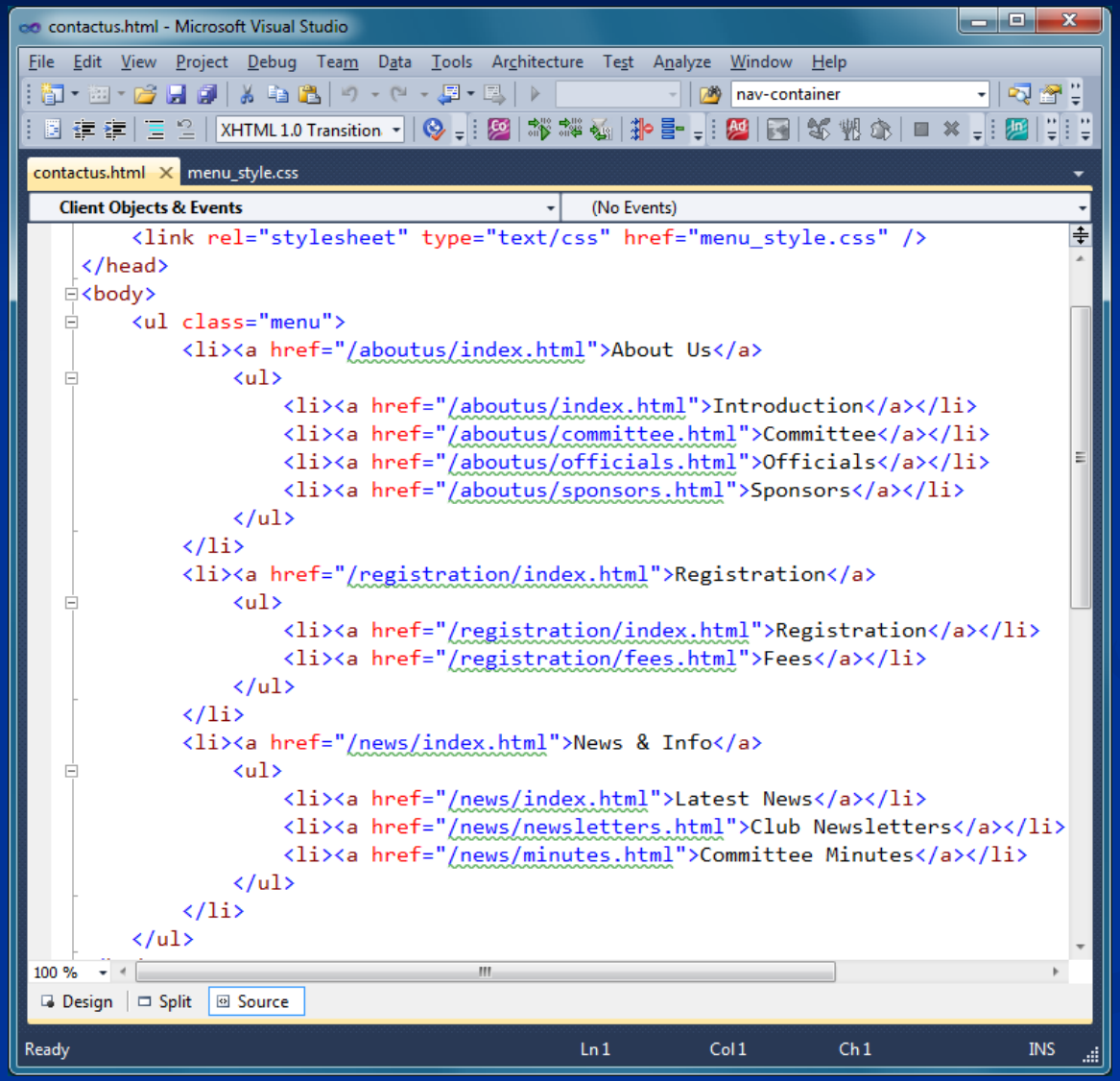
- Setting the left and right margin properties to **`auto`** has the effect of centering the box within its container:

```
margin-left: auto:
margin-right: auto;
width: 600px;
height: 800px;
background-color: red;
```

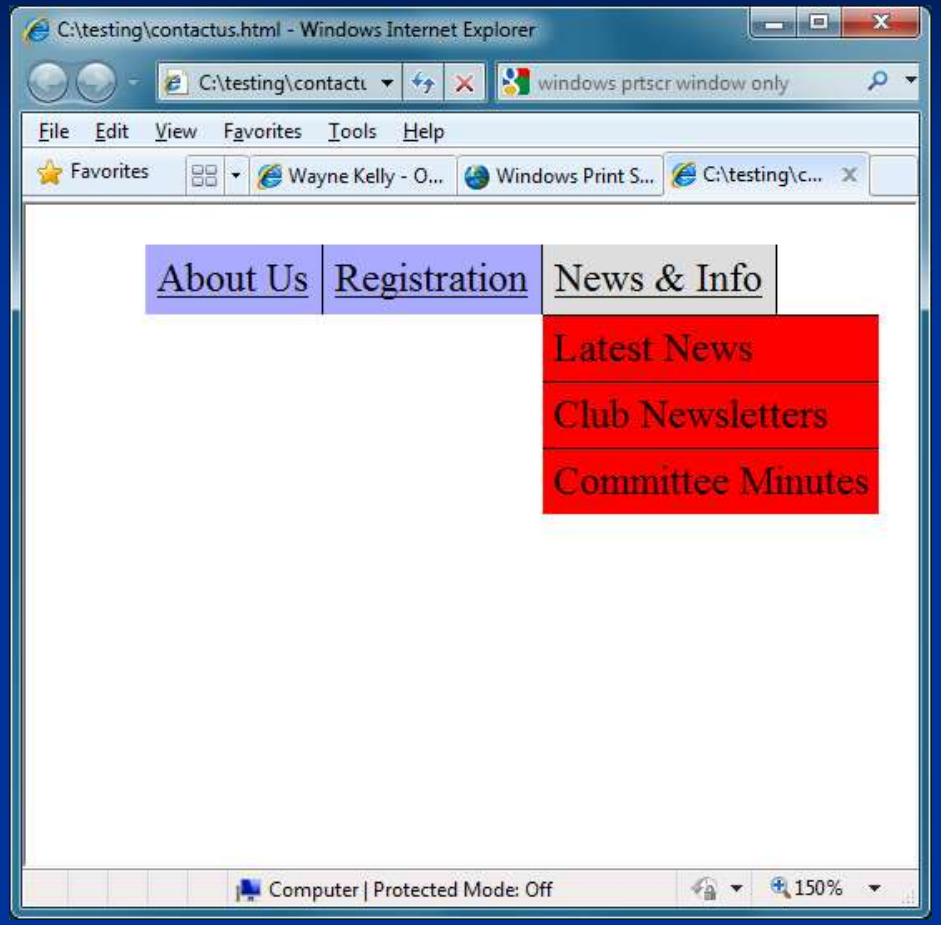


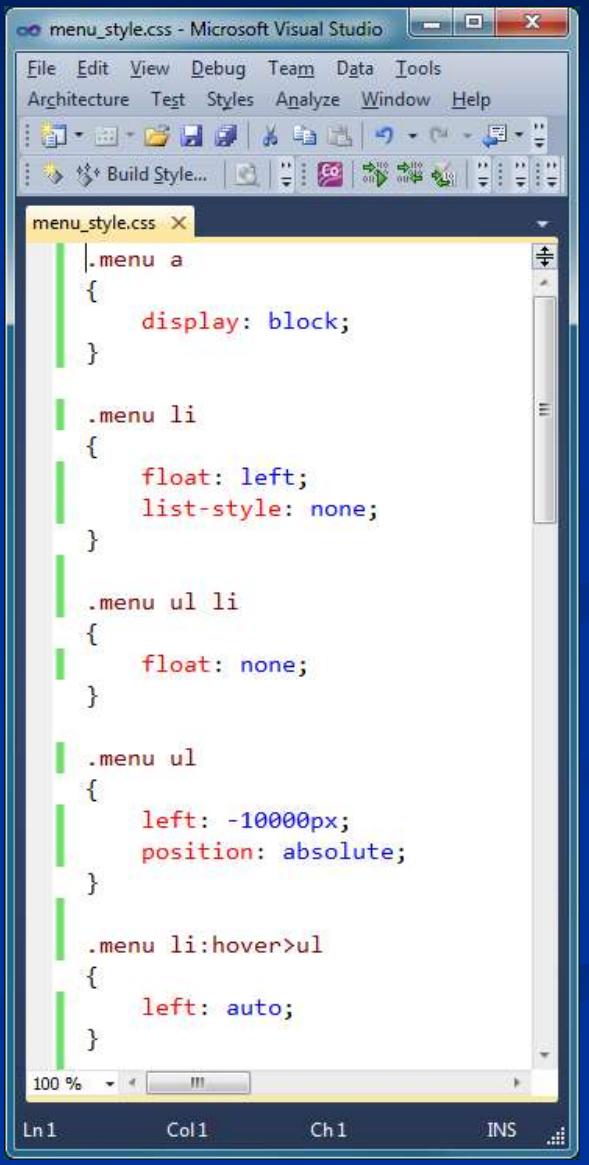
- **This works even if the browser window is resized**
  - **good for implementing *centred* page designs for dealing with low resolution monitors.**

CSS Drop Down Menus



Same Menu with CSS styling





- Example site
- Tutorial

CSS References

Tutorials:

http://www.htmlhelp.com/reference/css

http://www.w3schools.com/css

Specification:

http://www.w3.org/Style/CSS

http://www.w3.org/TR/REC-CSS1

http://www.w3.org/TR/REC-CSS2

Web Content Accessibility Guidelines

- The Web Content Accessibility Guidelines (WCAG) documents explain how to make Web content accessible to people with disabilities.

- Web Content Accessibility Guidelines (WCAG) Overview
- How to Meet WCAG 2.0 (Quick Reference)
- WCAG 2.0 Technical Specification

HTML Forms

HTML Forms

- HTML documents can contain one or more Forms.
- Each form is delineated by:

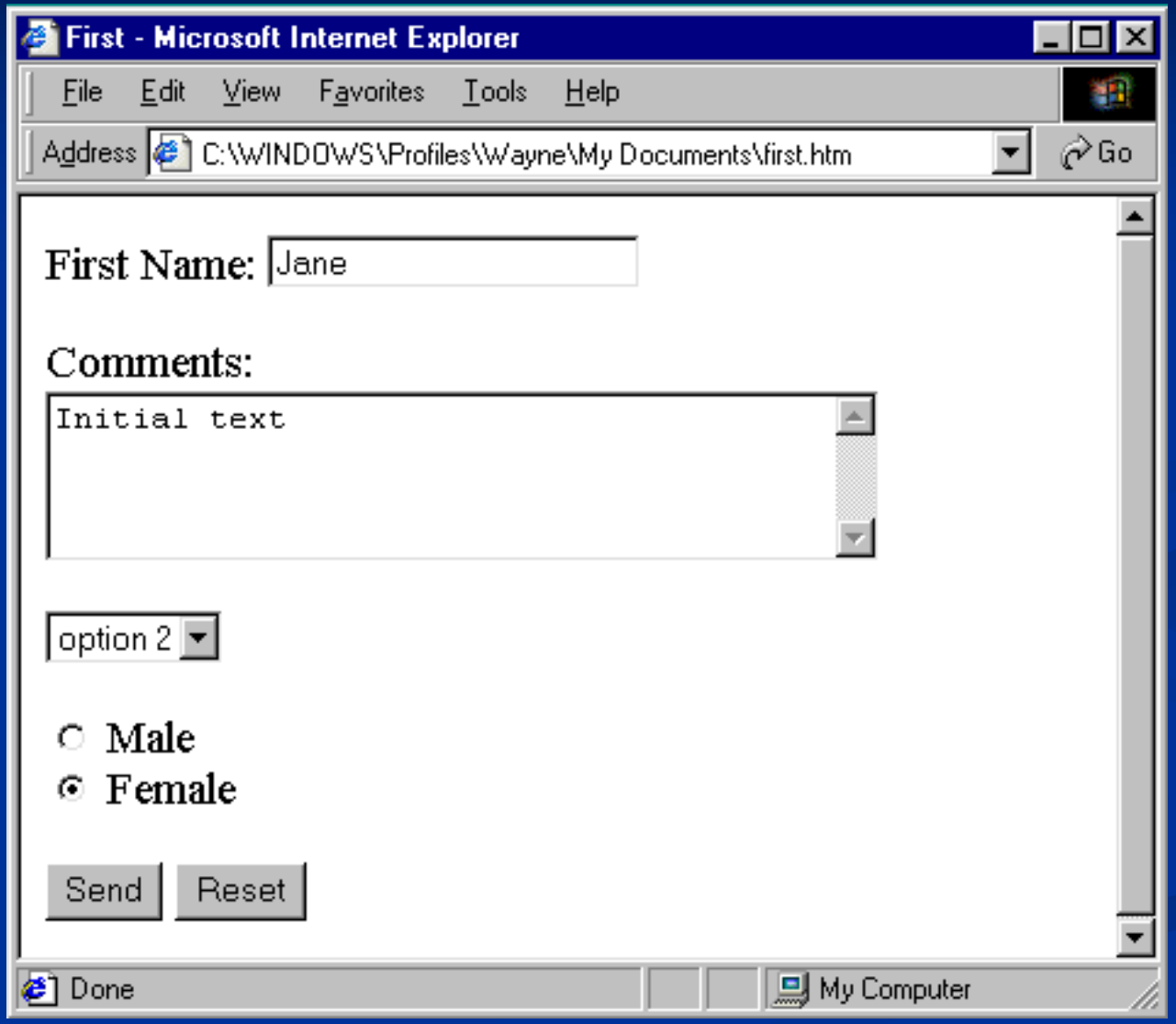
```
<form>
...
</form>
```

- Forms can contain normal HTML content just like other sections of the document.
- They can also contain one or more controls. Valid control types include:
  - `buttons`, `textboxes`, `checkboxes` and `menus`.
- Users *complete* a form by modifying its controls
  - entering text, selecting menu items, etc.
- The form is then generally *submited* to an agent for processing
  - e.g. to a Web server, to a mail server, etc.

HTML Form Controls

- Buttons
  - submit buttons `<input type="submit" ...>`
  - reset buttons `<input type="reset" ...>`
  - push buttons `<input type="button" ...>`
  - image buttons `<input type="image" ...>`
- Checkboxes `<input type="checkbox" ...>`
- Radio buttons `<input type="radio" ...>`
- File Select `<input type="file" ...>`
- Hidden controls `<input type="hidden" ...>`
- Text input `<input type="text" ...>`
- Password input `<input type="password" ...>`
- Multi-line input `<textarea ...>`
- Menus `<select ...> ... <option ...>`
- Object controls `<object ...>`

HTML Forms Example



HTML Forms Example

```
<form action="http://somesite.com/prog/adduser" method="post">
    First Name:
    <input type="text"   name="firstname" value="Jane" size="20">
    <input type="hidden" name="surname"   value="Doe">
<p>
    Comments:<br>
    <textarea name="thetext" rows="4" cols="40">
    Initial text
    </textarea>
<p>
    <select name="selection" size="1">
        <option          > option 1 </option>
        <option selected> option 2 </option>
        <option          > option 3 </option>
        <option          > option 4 </option>
    </select>
<p>
    <input type="radio" name="sex" value="Male"           > Male<br>
    <input type="radio" name="sex" value="Female" checked> Female
<p>
    <input type="submit" value="Send">
    <input type="reset">
</form>
```

HTML Forms References

Tutorials:

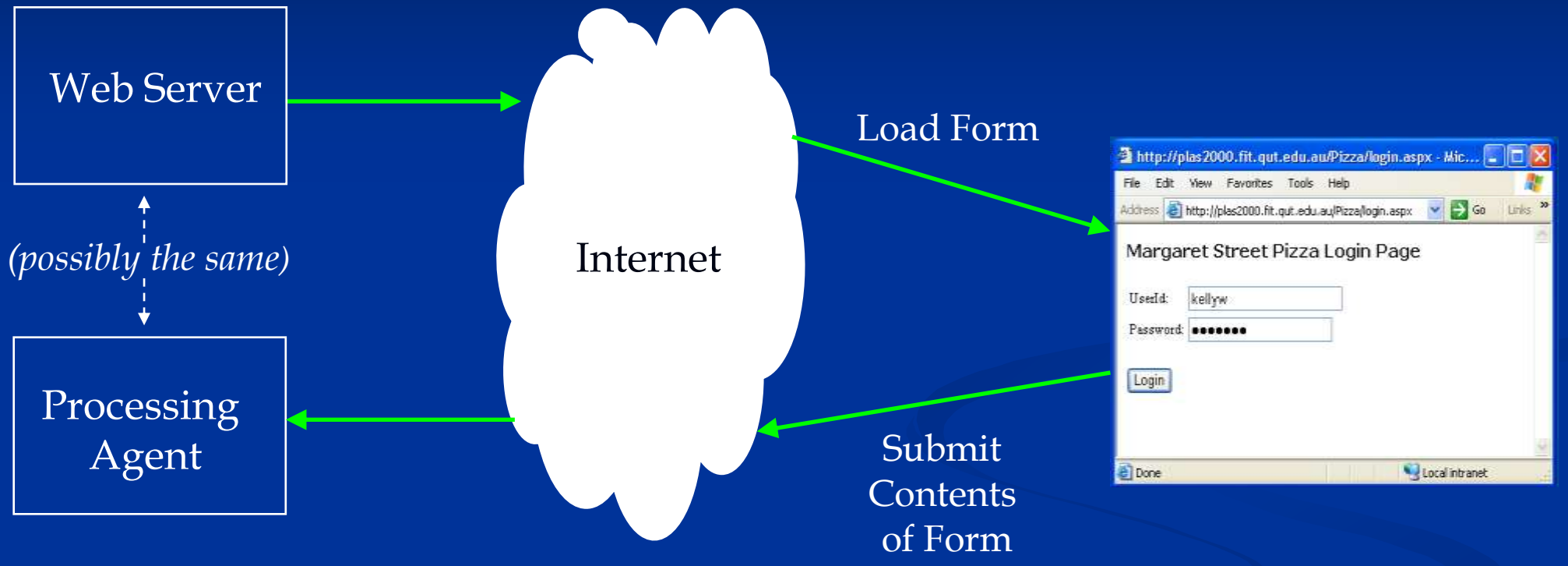
http://www.webcom.com/html/tutor/forms/start.shtml

http://www.davesite.com/webstation/html/chap17.shtml

Specification:

http://www.w3.org/TR/html4/interact/forms.html

Loading and Submitting HTML Forms



Form Submission

- `<form name="formname" method="get|post" action="URI">`
- When the user presses the form's submit button, the "contents" of the form is sent to an agent at the specified URI.
- The contents of the form consists of a (name, value) pair for each "successful" control on the form.
- The manner in which this content is passed to the server depends on the form's `method` attribute.
- If `method="get"`, the form's content is appended to the requested URI, eg:

```
http://somesite.com/prog/adduser?comment=Hello+world&selection=option+4&sex=Male
```

- If `method="post"`, the form's content is sent as the body of the URI request.

```
POST http://somesite.com/prog/adduser/prog/adduser HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: 47

comment=Hello+world&selection=option+4&sex=Male
```

HTML Event Handling

- Client-side scripts are commonly used to respond to form events such as a button being pressed, or a text box being changed.
- Event handlers implemented using scripting languages can be attached to controls and other html elements as follows:

```
<INPUT type="button" value="label" onclick=myproc>
```

- Event that can be handled include:
  - `onload, onunload, onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout, onfocus, onblur, onkeypress, onkeydown, onkeyup, onsubmit, onreset, onselect, onchange`

Client-side Scripting

JavaScript

Client-Side Scripting Languages

- The HTML specification allows any client-side scripting language to be used:

```
<SCRIPT TYPE="script-language" SRC="http://www.acme.com/scripts/bar.fs" >
</SCRIPT>
```

- The set of languages recognized varies between browsers:
  - Netscape: JavaScript and Tcl (plug-in)
  - Internet Explorer: VBScript and JScript (a.k.a. JavaScript)
- A short history of JavaScript (a.k.a. ECMAScript)

1995 LiveScript created by Netscape
LiveScript renamed JavaScript
1996 VBScript released by Microsoft
JScript - Microsoft's port of JavaScript
1997 ECMAScript (European Computer Manufacturers Association)
1998 ECMAScript became ISO standard (16262:1998)

Using Scripts with HTML Forms

- Client-Side scripts are commonly used to:
  - keep controls in “sync” with one another, and
  - to validate form content before passing it back to the server.
- Client-side scripts can:
  - access and modify the state of controls on forms.
  - popup standard alert and dialog boxes.
  - open, close and customize new browser windows.
  - modify the appearance of the document as it's loaded.
  - save and retrieve cookies from the clients computer.
- They can't, however, directly access server-side resources such as databases.

HTML Forms JavaScript Example

```
<script type="text/javascript">
     function validate(form) {
         var studnum = form.studnr.value
         if (studnum.length != 8){
             alert("Invalid Student Number - not enough digits")
             form.studnr.focus()
             return false
         }
         if (studnum.charAt(0) != 'n') {
             alert("Invalid Student Number - must start with 'n'")
            form.studnr.focus()
             return false
         }
         for (var i=1; i< studnum.length; i++) {
             if ((studnum.charAt(i) < '0') || (studnum.charAt(i) > '9'))
             {
                 alert("Invalid Student Number - illegal characters")
                 form.studnr.focus()
                 return false
             }
         }
         if (form.option.selectedIndex == 0) {
             alert("You must select a GUI development option")
             form.option.focus()
             return false
         }
     return true
     }
</script>
```

JavaScript: Objects but not Classes!

- JavaScript is object-based, but not object-oriented.
  - It supports objects, but not classes (uses prototype objects instead).
- It is loosely typed, which means:
  - you don't have to specify a data type when you declare a variable.
  - Any value can be assigned to any variable.
- Properties and Methods can be dynamically added and removed from individual objects:

```
function CircleArea()
{
  return Math.PI * this.Radius * this.Radius;
}

MyCircle = new Object();
MyCircle.Radius = 4.3;          // defines a new property
MyCircle.Area = CircleArea;     // defines a new method
MyCircle.Area();                // invokes new method
```

Constructing Objects in JavaScript

- "Constructor" functions make it easier to initialize new objects:

```
function Person(str, i)
{
   this.name = str;
   this.age = i;
}
fred = new Person("Fred Smith", 34); // new creates new object first
```

- Objects can inherit from other objects (prototypes).
  - the new object inherits the prototype's state as well as its methods.

```
function Student(str, i, j)
{
   this.Person(str, i);
   this.id = j;
}
Student.prototype = new Person; // a property of the Student constructor function
richard = new Student("Richard Sneddon", 35, 21351134);
```

Dynamic changes to a prototype object affects all objects constructed using it:

```
Student.prototype.Print = PrintStudent;
Person.prototype.species = "Human";
richard.Print();
```

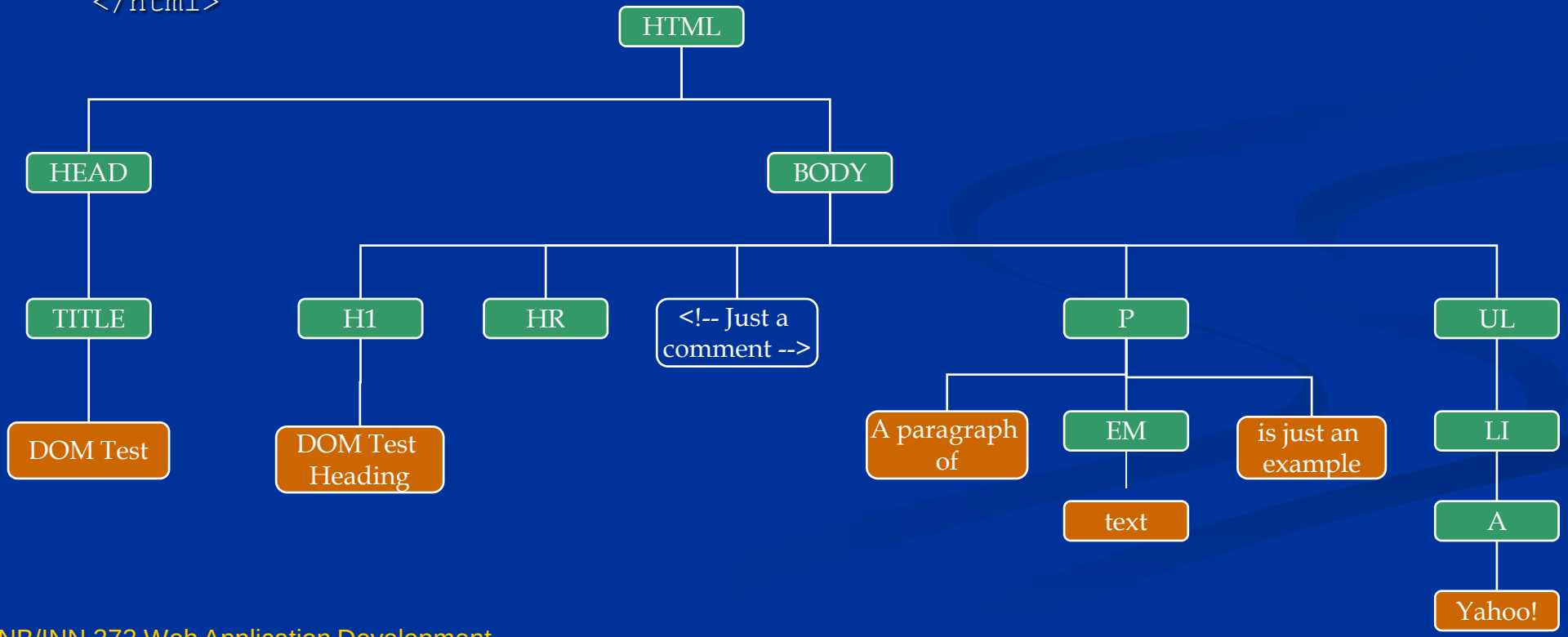
"Standard" JavaScript Objects

- Built-in (intrinsic) objects include:
  - `Array`, `Boolean`, `Date`, `Function`, `Global`, `Math`, `Number`, `Object` and `String`.
- Host environments such as Web Browsers and Web Servers provide additional objects to JavaScript programs.
  - These objects are not part of standard JavaScript and can be accessed by other scripting languages.
- Each Browser has traditionally had it's own object hierarchy for exposing the state of the Browser to programmers (DHTML APIs).
  - This made it extremely difficult to write browser independent scripts.
- In 1998, the W3C released a standard *Document Object Model* (DOM) for programmatically accessing the contents of HTML and XML documents.
  - Netscape 6 and Internet Explorer 5+ "support" the W3C DOM (Level 1).

DOM

Document Object Model

```
<html>
    <head><title>DOM Test</title></head>
    <body>
        <h1>DOM Test Heading</h1>
        <hr>
        <!-- Just a comment -->
        <p>A paragraph of <em>text</em> is just an example</p>
        <ul>
            <li><a href="http://www.yahoo.com">Yahoo!</a></li>
        </ul>
    </body>
</html>
```



Using the DOM in Web Browsers

- Traditionally, scripting only allowed HTML documents to be modified as they were loaded.
- DOM allows scripts to modify HTML documents even after they have been loaded (Dynamic HTML).
- With DOM, any HTML element can be accessed, modified, removed or inserted using properties and methods such as:

Properties:

```
nodeType
parentNode
childNodes
firstChild
lastChild
previousSibling
nextSibling
attributes
```

Methods:

```
getElementById
createElement
createAttribute
createTextNode
insertBefore
replaceChild
removeChild
appendChild
```

DOM Example

```
<script>
    function Insert()
    {
        input = document.getElementById("InputWord").value;
        newPoint = document.createElement("LI");
        newText = document.createTextNode(input);
        newPoint.appendChild(newText);
        list = document.getElementById("MyList");
        child = list.firstChild;

        while (child != undefined)
            if (child.firstChild.nodeValue > input)
                break;
            else
                child = child.nextSibling;

        if (child != undefined)
            list.insertBefore(newPoint, child);
        else
            list.appendChild(newPoint);
    }
</script>
<form ID="Form1">
    <input type="text" id="InputWord" NAME="InputWord">
    <input type="button" value="Insert" onclick="Insert()">
</form>
<ul id="MyList">
    <li>
        Point 1
    </li>
</ul>
```

DOM References

Introduction:

http://www.w3schools.com/HTMLDOM/

Specification:

http://www.w3.org/DOM/

AJAX
Asynchronous JavaScript and XML

- Designed to make web pages more interactive and responsive (without reloading entire page).
- Browser uses XMLHttpRequest object to asynchronously communicate small amounts of (usually) XML data with the server.
- Uses DOM and JavaScript to dynamically update the document.

JavaScript Libraries

- http://javascriptlibraries.com/
- 40 Useful JavaScript Libraries for specific purposes
- Common Features:
  - Hide browser dependencies
  - Widgets
  - AJAX support
- Popular Examples:
  - jQuery
  - Dojo
  - Prototype
  - Yahoo UI
- Also server-side frameworks which generate JavaScript

JavaScript References

Introduction:

http://devedge-temp.mozilla.org/library/manuals/2000/javascript/1.5/guide/

http://msdn.microsoft.com/library/en-us/dninstj/html/languagetour.asp

http://www.devguru.com/Technologies/ecmascript/quickref/javascript_intro.html

Specification:

http://www.ecma-international.org/publications/standards/Ecma-262.htm

Cookies

- A small piece of text stored on a user's computer by a web browser
  - name-value pairs, pure text - not executable
  - used for authentication, session tracking, personalization
- Web server can send a cookie to a web browser via a header field of a HTTP response message
- Browser should then automatically send this cookie in subsequent HTTP request messages to that web server (until expired).
- Can be disabled or removed by user
  - Many web sites won't function correctly if cookies are disabled
- Can be read and written by client-side scripts
  - `document.cookie`

Testing Web Sites

Browser versions:

- Internet Explorer 6, 7, 8, 9
- Firefox 3, 4
- Chrome: 6, 7, 8, 9
- Safari: 4, 5
- Opera: 9, 10, 11

Screen resolutions:

- 600 x 800 ?
- 1024 x 768
- Higher

Java Applets, Flash, ActiveX and .NET Client-Side Controls

- Java Applets, Flash, ActiveX and .NET Controls can be linked into HTML documents using the OBJECT tag:

```
<OBJECT classid="java:Bubbles.class"
        codebase="http://foo.bar.com/java/myimp/"
        width="500" height="500">

 <OBJECT width="550" height="400">
        <param name="movie" value="somefilename.swf">
        <embed src="somefilename.swf" width="550" height="400">
        </embed>
 </OBJECT>

 <OBJECT classid="clsid:663C8FEF-1EF9-11CF-A3DB-080036F12502"
        codebase="http://foo.bar.com/activex/myimp/ctrl.ocx"
        width="500" height="500">

 <OBJECT classid="VolunteerHost.dll#G2.VolunteerGUI"
        width="500" height="500">
```

- All depend on appropriate plug-in and OS support
  - won't work for many users
  - may be disabled by users