CSS, JAVASCRIPT and DHTML

Qiang Yang
Simon Fraser University
Thanks: F. Lau (HKU) and E. Cerami (NYU)

Structure vs. Style

- HTML is a **presentational language**
- A document contains both **structural** and **presentational** (or **stylistic**) information
  - Structure: headers, sections, paragraphs, footers,
  - Style: fonts, color, indentation, spacing, etc.
- In the style sheet approach, the structural information is separated from the presentational information
  - The latter is contained in a **style sheet**

Structure and Format

- We have seen some examples of structure vs. styles
  - In Latex:
    - Styles: \newcommand{\proof}{{\bf Proof}}
    - Structures: \section{}, \title{},
- Style sheet along with JavaScript and other technologies make up what is now widely known as **Dynamic HTML** (**DHTML**)

Style Sheets

- Inline
- Declared
- Imported

CSS Defined

- CSS: a simple mechanism for adding style (e.g. fonts, colors, spacing) to Web documents.
- Two Step process for using CSS:
  - Step 1: Create your "styles"
  - Step 2: Apply your styles to your HTML document.
- Let's look at an example…


```
<HTML>
<BODY>
<STYLE TYPE="text/css">
<!--
.HEADING1 {
        color: blue;
        font-size: 64px;
}
.HEADING2 {
        color: gray;
        font-size: 22px;
}
-->
</STYLE>
<SPAN CLASS="HEADING1">Resume Posting Service</SPAN>
<P>
<SPAN CLASS="HEADING2">Provided by hotcomputerjobs.com</SPAN>
</BODY>
</HTML>
```

First, you create your styles Within a <STYLE> tag.

Then, you apply your styles By using the SPAN tag.

Style Sheet Example

- An example rendered
- This example contains a style sheet marked by the tags <style></style>
- A style sheet contains a number of **rules**
- The following rule applies to the BODY

```
BODY {
      font-family: "times new roman", times, serif;
      background: white;
      color: black;
      margin-left: 10%;
      margin-right: 10%;
      text-align: justify;
}
```


Defining Styles

- Each Style has a name, and a set of properties.
- For example, the heading1 tag is set to blue, 64 pixels big:

```
.HEADING1 {
    color: blue;
    font-size: 64px;
}
```

- Lots of properties exist: color, font-size, text-align, font-family, etc.

Applying Styles

- Once you have created your styles, you apply a style to your text via the SPAN tag.
- For example, to apply the heading1 style:

```
<SPAN CLASS="HEADING1">Resume Posting Service</SPAN>
```


Span and Div

- **SPAN** element: generic grouping element
  - Does not apply any inherent formatting
  - Main use is to apply styles or **ID** attributes to block of text
  - *Inline element*
- **DIV** element
  - Similar to **SPAN**, but *block-level element*
    - Displayed on own line with margins above and below

Cascade Precedence Rules

- There are many ways styles can be applied to an element
  - <style> tag set
  - External style sheet file
  - STYLE attribute in a tag
- There is the possibility that multiple style rules can apply to the same element in a document
- The CSS recommendation has a set of rules for resolving conflicts
  - Inline style attributes have precedence over internal style sheet which has precedence over external style sheet

Dynamic HTML with JACASCRIPT

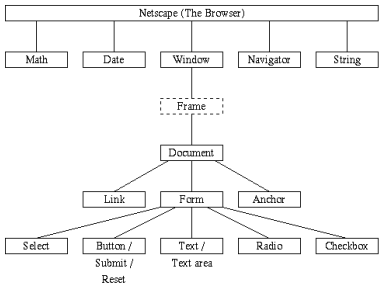
Client-Side Activeness

- "Dynamic HTML is the combination of **HTML**, **style sheets** and **scripting** used to create an interactive experience for the reader without additional server interaction, CGI scripts, or other network communication"
- Client-side scripting, client-side activeness
  - Without DHTML, the browser must download another page from the server to change what the user sees on the screen

Document Object Model (DOM)

- When an HTML page loads into a scriptable browser, the browser creates a hidden, internal roadmap of all the elements it recognizes as scriptable objects
- The roadmap is hierarchical
- The scriptable objects are "document objects"
- Document objects in a page can be addressed and moved around, font sizes and styles can change as the cursor travels over them … all this is controlled by scripting
- Unfortunately, Netscape and Microsoft have somewhat incompatible implementations of DHTML

DOM (cont'd)



JavaScript

- JavaScript (or Microsoft's **JScript**) is the main, most popular vehicle for client-side activeness
- History: LiveScript (before NN2), JavaScript (NN2, 1995), ECMAScript (W3C, ongoing)
- JavaScript is directly interpreted (no byte-code), object-oriented, C-like, untyped
- Java **applets** are also for client-side activeness, but they are more clumsy, less akin to the browser
- Both Java applets and JavaScript let "executable contents" to be embedded in an HTML document

Examples in DDN Book, Ch15

- 01: Onload event triggers start()
  - alert("msg");
  - OnLoad="start()";
- 02: for-loop using object model
  - document.all.length
  - Document.all[index].tagName
  - pText.innerHTML

Examples in DDN Book: CSS, JAVASCRIPT with MM Active-X Objects

- Spinning Earth
- Flying earth

More Examples

- Some simple examples
  - Fibonacci
  - onClick
  - Text colors
- More examples
  - Netscape Dynamic HTML Developer's Guide by Harris & Kidder (IE has problems)
  - The Dynamic Duo Cross-Browser DHTML (with tutorial)

JavaScript vs. Java

- The two are entirely unrelated, except some syntactic similarities
- The two languages have disjoint sets of capabilities:
  - JavaScript can control browser behavior and content but cannot draw graphics or perform networking;
  - Java has no control over the browser, but can do graphics, networking, and multithreading

JavaScript Objects

- JavaScript is object-oriented
  - Everything is an object, which is an instance of some class
  - An object contains properties (variables, references to other objects, etc.) and methods
- For example, the "window object" contains two properties which are references that refer to the window object itself: window, self
  - The following are equivalent
    - `window.alert("oops!");`
    - `alert("oops!");`
    - `self.alert("oops!");`

Navigator, Location, and History

- The Window object contains references to three objects that contain information about the browser or the browser window itself: the Navigator object, the Location object, and the History object
- An example of determining the browser version information through the Navigator object's properties
- This one enumerates all the properties in the Navigator object and the names of all the plugin's installed and supported MIME types