HTML 5

This lecture is based on materials from:

HTML5: The Missing Manual
The Book That Should Have Been in the Box
Matthew MacDonald
O'Reilly Media

Document Structure

```
<!DOCTYPE html>
<html lang="eng">
  <head>
        <title>The page's title</title>
        <meta charset="utf-8">
        <link href="mystylesheet.css" rel="stylesheet">
        <script src="myscript.js"> </script>
  </head>
  <body>
        <p lang="fr">Ceci est un paragraphe.</p>
  </body>
</html>
```

Syntax

- <html>,<head>, and <body> are optional
- Ignores capitalization
  - <P>This is a <EM>fine</eM>example</p>
- OK to omit closing slash from void element
  - Both
 and
 are ok
- Attribute values don't need quotation marks
- Value-less attributes are allowed
  - <input type="checkbox" checked>

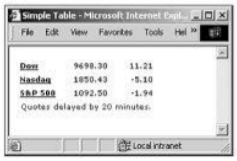
Good Style

- Include <html>,<head>, and <body> tags
- Use lowercase tags
- Use quotation marks around attribute values

Cascading Style Sheets (CSS)

5

Bad Idea: Mixing Presentation and Document Structure



2,400 HTML characters to describe 60 characters of content

6

Cascading Style Sheets (CSS)

- Separate structure from presentation
- "Simple" mechanism to attach style to structured documents
  - fonts, colours, spacing, …

7

CSS Advantages

- Precise control over presentation
- Simplify site maintenance
- Faster downloads
- Media-specific rendering

8

CSS Language

stylesheet: ruleset*

ruleset: selector '{' [declaration ';']* '}'

declaration: property ':' expr ['! important']?

```
p {
  font-family: sans-serif;
  color: red;
}
```

9

Selectors

- Type E
- Universal *
- Grouping E,G,F
- Attribute [foo="hi"]
- ID #myID or E#myID
- Class .myClass
- Pseudo-element E:pseudo-element
- Contextual
  - Descendent E F
  - Child E > F
  - Adjacent E + F

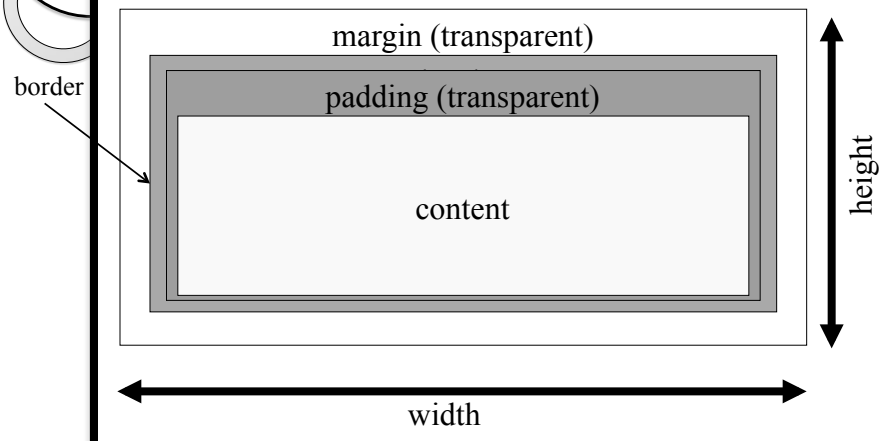
10

Available Formatting

- Font
- Text
- Background
- Display
- Box
- Positioning
- Animation

11

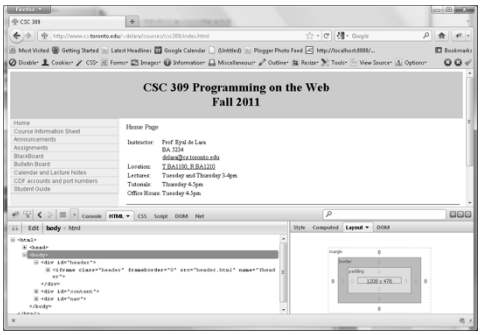
Box Formatting Model



12

Firebug: CSS debugging

- Firefox add-on (http://getfirebug.com/)
- Inspect XHTML/CSS of any page
- Change styles dynamically

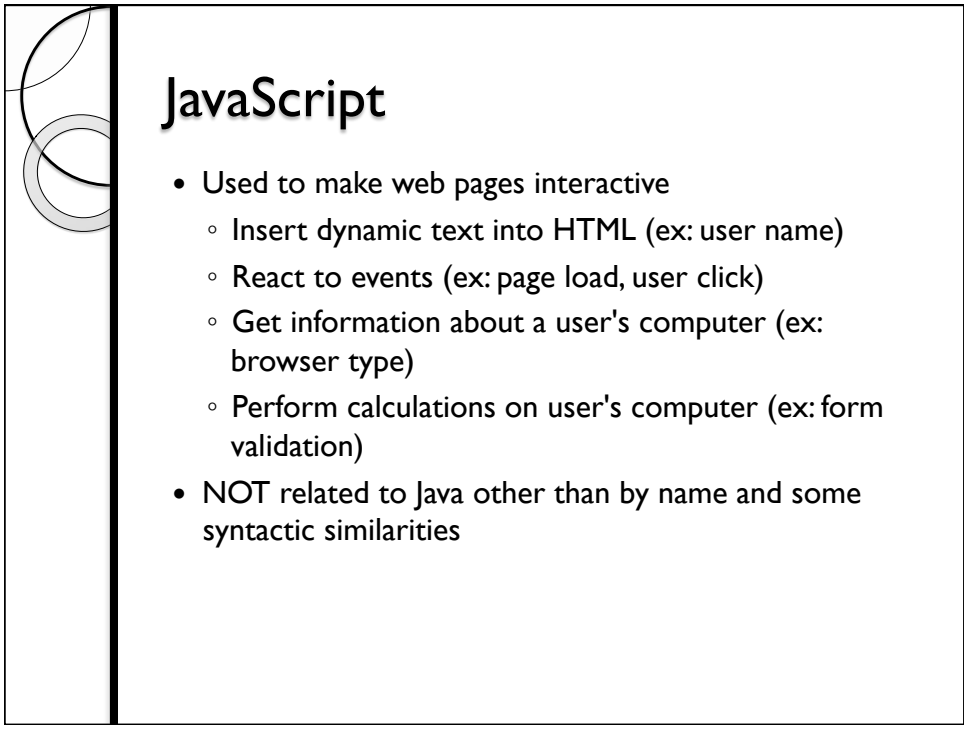


Javascript

This lecture is based on materials from:

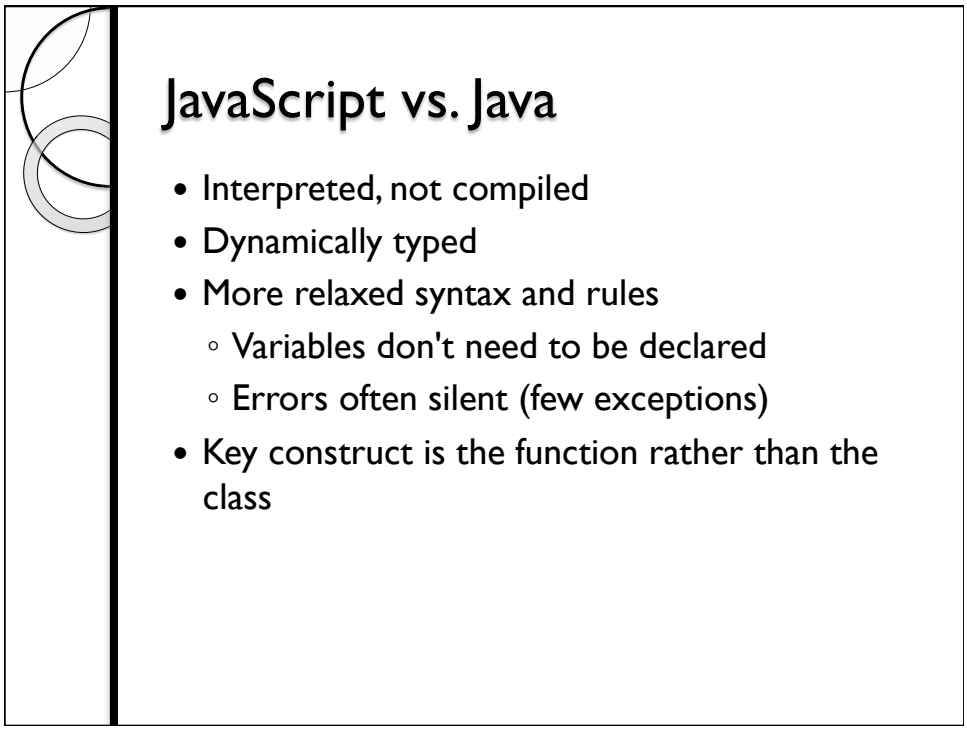
Eloquent JavaScript
A Modern Introduction to Programming
by Marijn Haverbeke

http://eloquentjavascript.net/



JavaScript

- Used to make web pages interactive
  - Insert dynamic text into HTML (ex: user name)
  - React to events (ex: page load, user click)
  - Get information about a user's computer (ex: browser type)
  - Perform calculations on user's computer (ex: form validation)
- NOT related to Java other than by name and some syntactic similarities



JavaScript vs. Java

- Interpreted, not compiled
- Dynamically typed
- More relaxed syntax and rules
  - Variables don't need to be declared
  - Errors often silent (few exceptions)
- Key construct is the function rather than the class

JavaScript Security

- Language/API limitations:
  - No file/directory access defined in the language
  - No raw network access. Limited to either
    - load URLs
    - send HTML form data to
      - web servers, CGI scripts, e-mail addresses
  - 'same origin policy'
    - can only read props of documents and windows from the same place: host, port, protocol
- Privacy restrictions:
  - cannot read history
  - cannot hide/show menubar, status line, scrollbars cannot close a window not opened by itself

Variables

- Declaration
  - Explicit var i = 12; // no type declaration
  - Implicit msg = "hello";
- Name
  - Cannot start with a digit or include spaces
  - Examples:
    - catch22
    - $
    - $_

Dynamic Typing

- Different than Java or C
- Variables can hold any type of value:
  - number (64 bit floating point)
    - 144, 9.81, 2.99e8
  - string
    - 'You ain\'t seen nothing yet!'
  - Boolean
    - FALSE: "", null, undefined, NaN, false
    - TRUE: everything else (e.g., true, "hi", -1, 3.5)
  - function (first-class data type)
  - object
  - string
  - undefined
- … and can hold values of different types at different times during execution

```
var somevariable = 0;
somevariable = "new value";
somevariable = {2,'hi!',3.1415};
```

Operators

- Arithmetic
  - + - * / %
- Logic
  - && || !
- Comparison
  - < > == != <= >= === !==
- Other
  - typeof

Control and Looping

- Control
  - if
  - switch
- Looping
  - for
  - while
  - do..while
  - for .. in
    - for (*property* in *object*) {}

Embedding in HTML

- Directly

```
<script>
  ……..
</script>
```

- Indirect

```
<script src="test.js" />
```

Example

```
<!DOCTYPE html>
<html lang="eng">
    <head>
        <title>Loop</title>
        <meta charset="utf-8”>
        <script>
            var theNumber = Number(prompt("Factorial of?"));
            var count = 1;
            var factorial = 1;
            while (!isNaN(theNumber) && count <= theNumber) {
                        factorial *= count++;
                        console.log(factorial);
            }
        </script>
    </head>
</html>
```

Functions

```
function functionName ([arg1] […,argN])
{
        ……….
        [return [value]];
}
```

- Arguments
  - Primitive types (number, boolean) are passed by value
  - Object types are passed by reference

Example

```
<!DOCTYPE html>
<html lang="eng">
    <head>
        <title>Function Example</title>
        <meta charset="utf-8">
        <script>
            function factorial(num) {
              if (isNaN(num) || num ==0)
                    return 1;
              return num * factorial(num-1);
            }
            console.log(factorial(Number(prompt("Factorial of?"))));

        </script>

    </head>
</html>
```

Evaluation and Execution

- Evaluation
  - As document is parsed, in order
- Execution
  - Statement outside functions
    - When it is encountered
  - Statement inside function
    - When function is called
      - Event handler

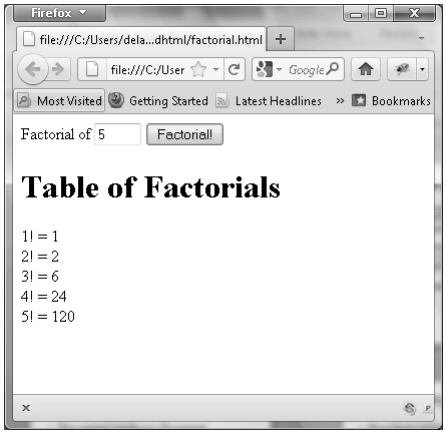
        <body onload="helloWorld()">

Document Object Model (DOM)

- W3C Standard
- Interface between document displayed by browser and application programs
- Platform-neutral and language-neutral collection of interfaces
- Documents have treelike structures
- Create documents, move around document structure (parse), and change, add, or delete elements.

Example: Factorial

- Print factorial table



Associating Events with Elements

- In the HTML
  - As value of attributes

```
<a href="…" onmouseover="popupFunc();" />
```

- In a script
  - Explicit reference to object's event handler

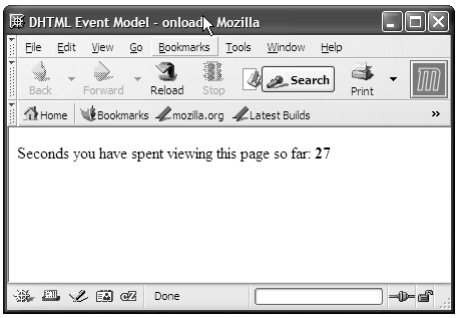
```
document.onmouseover = functionFoo;
```

onload & timers

- onload
  - Fires when element (an all children) finish loading
  - Used in the <body> to execute script after page has been rendered

Example: Onload & Times

- Example: Count how many seconds have passed since page finish rendering



Changing Style Attributes

- CSS is scriptable from JavaScript
  - allows HTML elements to float around and grow and shrink.

Tracking Mouse Movements

- Track mouse position on screen
- Drag and drop ball on click
- Events onmousemove and onclick