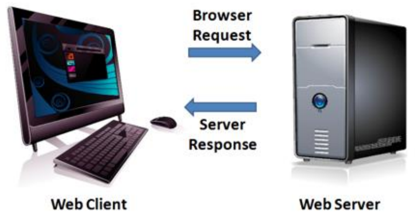
JavaScript and jQuery Course

Session 01

Client / Server Model

Client – Web Browser
Server – Web Server

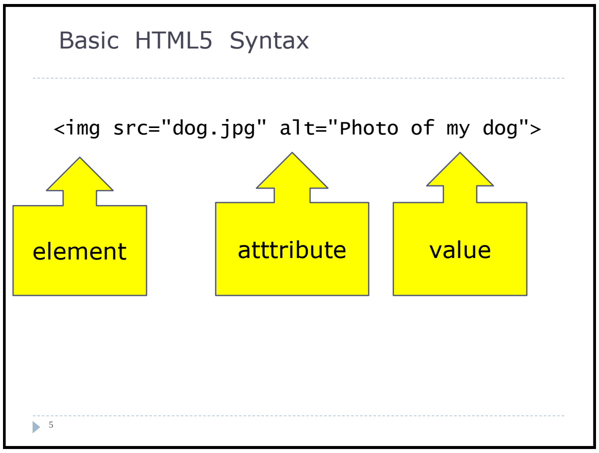


Client Sided Technologies

- Interpreted in browser
  - HTML
  - CSS
  - JavaScript
- Processed on Server
  - PHP
  - ASPX
  - JSP

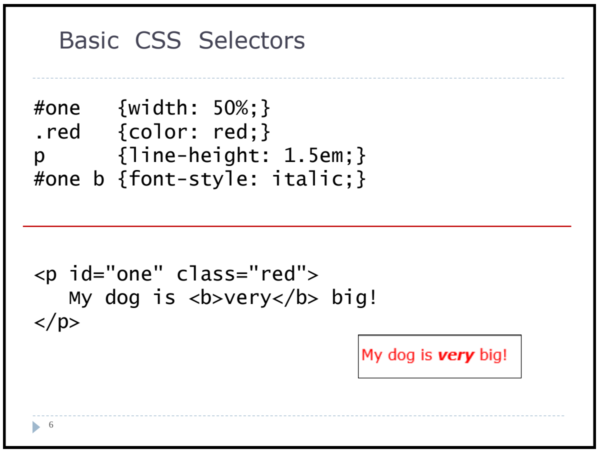
Example HTML5 Web Page

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Page Title Goes Here</title>
  </head>

  <body>
    ... body text and more HTML5 tags go here ...
  </body>
</html>
```



Basic HTML5 Syntax

```
<img src="dog.jpg" alt="Photo of my dog">
```

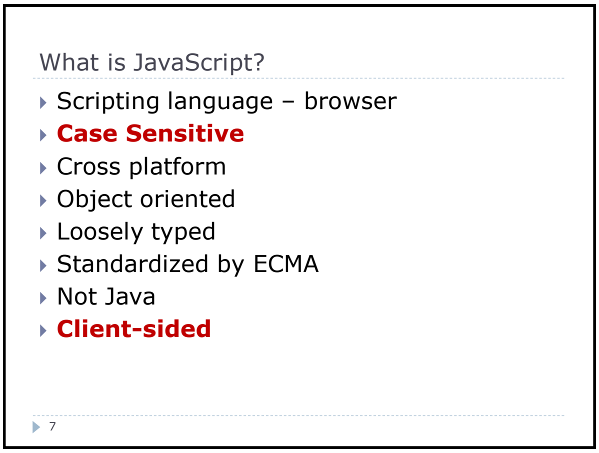


Basic CSS Selectors

```
#one   {width: 50%;}
.red   {color: red;}
p      {line-height: 1.5em;}
#one b {font-style: italic;}
```

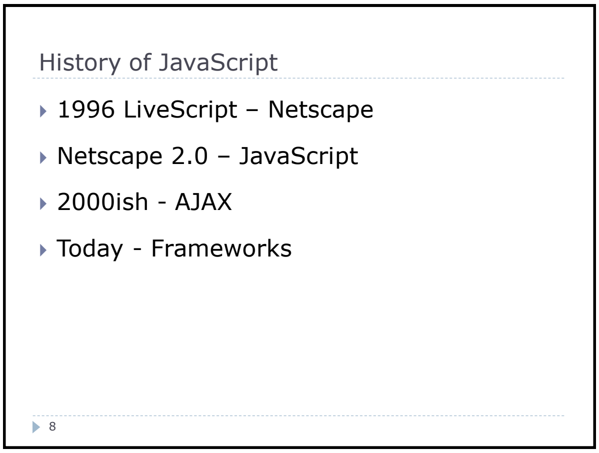
```
<p id="one" class="red">
   My dog is <b>very</b> big!
</p>
```

My dog is ***very*** big!



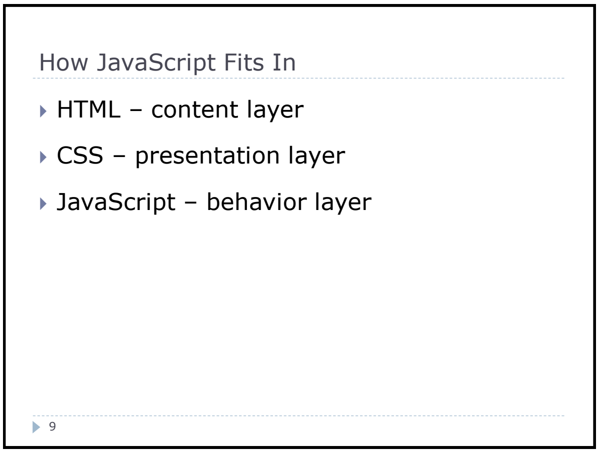
What is JavaScript?

- Scripting language – browser
- **Case Sensitive**
- Cross platform
- Object oriented
- Loosely typed
- Standardized by ECMA
- Not Java
- **Client-sided**



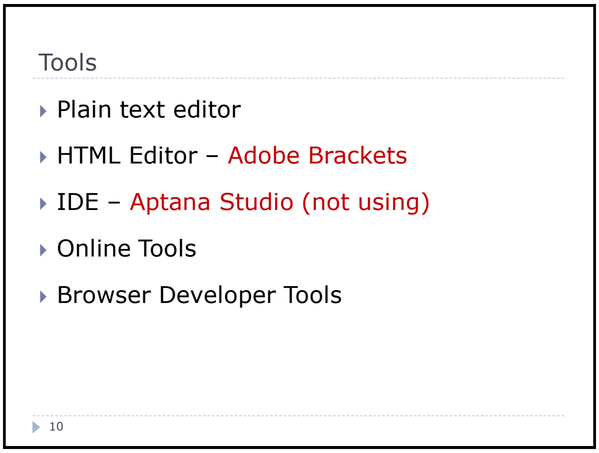
History of JavaScript

- 1996 LiveScript – Netscape
- Netscape 2.0 – JavaScript
- 2000ish - AJAX
- Today - Frameworks



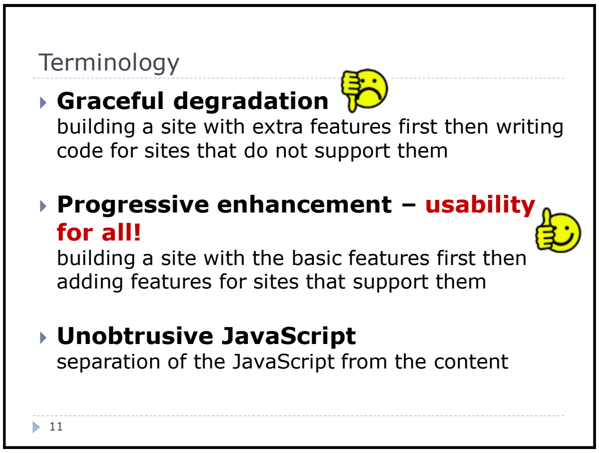
How JavaScript Fits In

- HTML – content layer
- CSS – presentation layer
- JavaScript – behavior layer



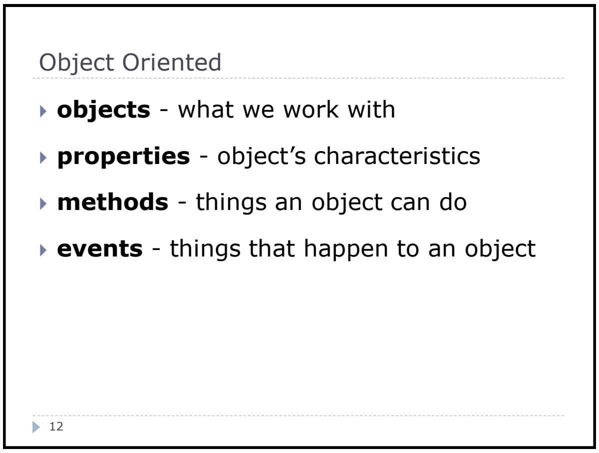
Tools

- Plain text editor
- HTML Editor – Adobe Brackets
- IDE – Aptana Studio (not using)
- Online Tools
- Browser Developer Tools



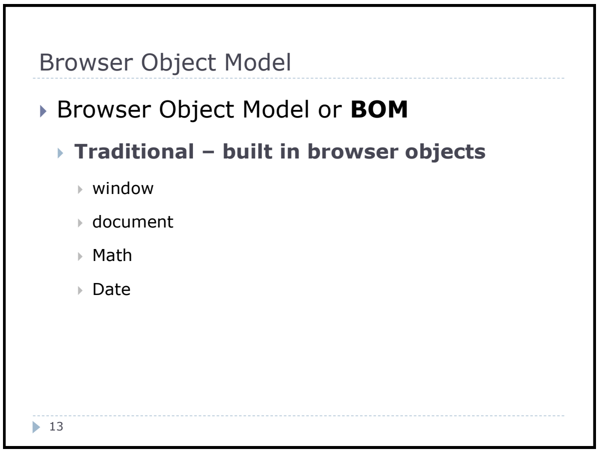
Terminology

- **Graceful degradation**
  building a site with extra features first then writing code for sites that do not support them
- **Progressive enhancement – usability for all!**
  building a site with the basic features first then adding features for sites that support them
- **Unobtrusive JavaScript**
  separation of the JavaScript from the content



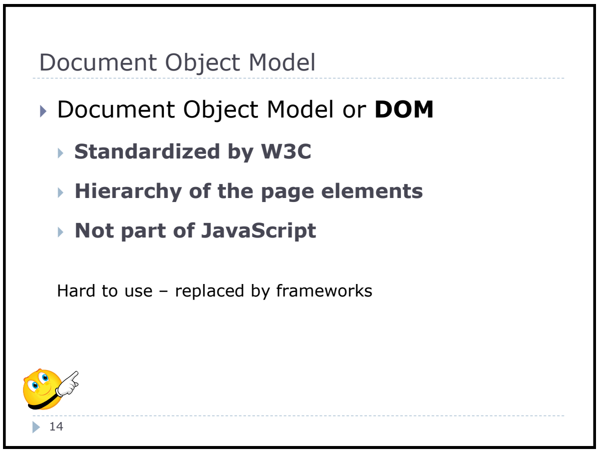
Object Oriented

- **objects** - what we work with
- **properties** - object’s characteristics
- **methods** - things an object can do
- **events** - things that happen to an object



Browser Object Model

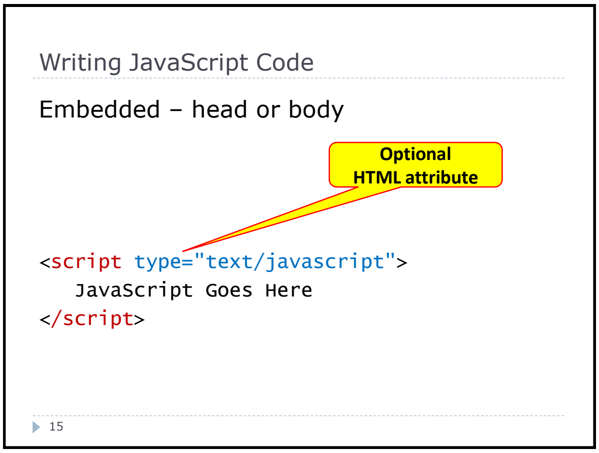
- Browser Object Model or **BOM**
  - **Traditional – built in browser objects**
    - window
    - document
    - Math
    - Date



Document Object Model

- Document Object Model or **DOM**
  - **Standardized by W3C**
  - **Hierarchy of the page elements**
  - **Not part of JavaScript**

Hard to use – replaced by frameworks

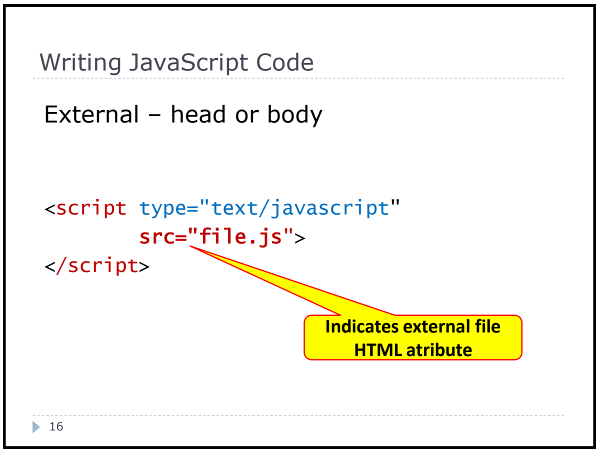


Writing JavaScript Code

Embedded – head or body

Optional
HTML attribute

```
<script type="text/javascript">
   JavaScript Goes Here
</script>
```



Writing JavaScript Code

External – head or body

```
<script type="text/javascript"
        src="file.js">
</script>
```

Indicates external file
HTML atribute

Writing JavaScript Code

For browsers that don’t support JavaScript

```
<noscript>
</noscript>
```

```
<h2>
    <noscript>
        To get the most from this web site,
        please enable JavaScript.
    </noscript>
</h2>
```

JavaScript Syntax – Comments

```
// single line

/*
   multiple line
   comment on
   several lines
*/
```

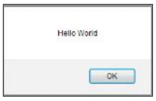
Comments – Example

```
/* this onload function sets up the events that display
   and hide the text that follows a series of h2 headings
*/
window.onload = function () {
    var fiveReasons = $("five_reasons"); // gets a div

    // gets the h2 and div elements within the div element
    var h2Headings =
        fiveReasons.getElementsByTagName("h2");
    var divTags = fiveReasons.getElementsByTagName("div");

    var i, headingNode, divNode;
    for (i = 0; i < h2Headings.length; i++ ) {
        // one loop for each h2
        headingNode = h2Headings[i];
        divNode = divTags[i];

        // Attaches an event handler for each h2
        headingNode.onclick = function () {
            var h2 = this;
```

JavaScript Syntax – How to write

- Dot syntax

```
window.alert()
```



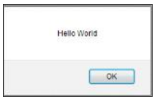
- Statements end in semicolon

```
alert("Hello World");
```

A method is followed by ( ) and may contain information

JavaScript Syntax – How to write

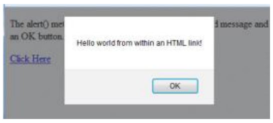
```
<head>
  <meta charset="utf-8">
  <title>The alert method of the window object</title>
  <script>
     alert("Hello World");
  </script>
</head>
```



Hello World

Traditional JavaScript – **Don’t Use**

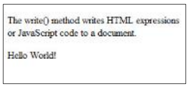
```
<body>
  <a href="#"
     onclick="alert('Hello world from within an HTML link!');
     return false;">Click Here</a>
</body>
```



The alert() met d message and an OK button.

Hello world from within an HTML link!

Traditional JavaScript – **Don’t Use**



The write() method writes HTML expressions or JavaScript code to a document.

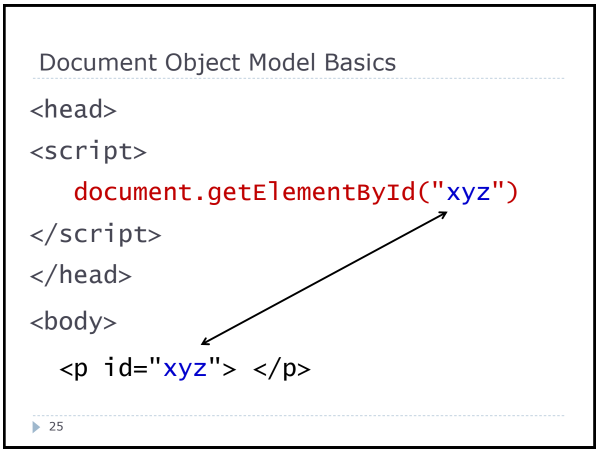
Hello World!

```
<body>

  <script>
     document.write("Hello World!");
  </script>

</body>
```

The `write` method of the document object

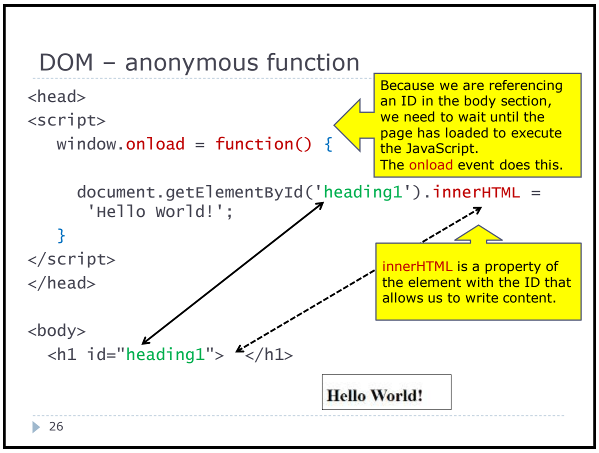
Document Object Model

- Allows us to write to the page using
  “unobtrusive JavaScript”
- Replaces `document.write()`



Document Object Model Basics

```
<head>
<script>
    document.getElementById("xyz")
</script>
</head>
<body>
   <p id="xyz"> </p>
```



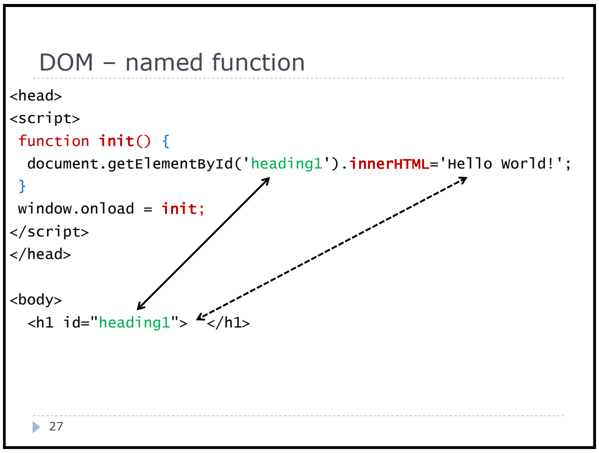
DOM – anonymous function

```
<head>
<script>
   window.onload = function() {

     document.getElementById('heading1').innerHTML =
      'Hello World!';
   }
</script>
</head>

<body>
  <h1 id="heading1"> </h1>
```

Because we are referencing an ID in the body section, we need to wait until the page has loaded to execute the JavaScript.
The onload event does this.

innerHTML is a property of the element with the ID that allows us to write content.

Hello World!



DOM – named function

```
<head>
<script>
 function init() {
  document.getElementById('heading1').innerHTML='Hello World!';
 }
 window.onload = init;
</script>
</head>

<body>
  <h1 id="heading1"> </h1>
```

How to create identifiers

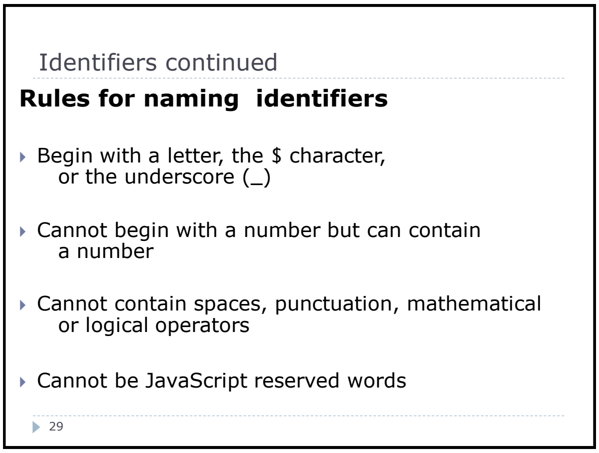
Identifier
The name we give to a variable, method, property, function, object

Camel casing syntax
All of the words, except the first word, start with a capital letter.

Example: `firstName`

Underscore notation
All of the words are joined by an underscore

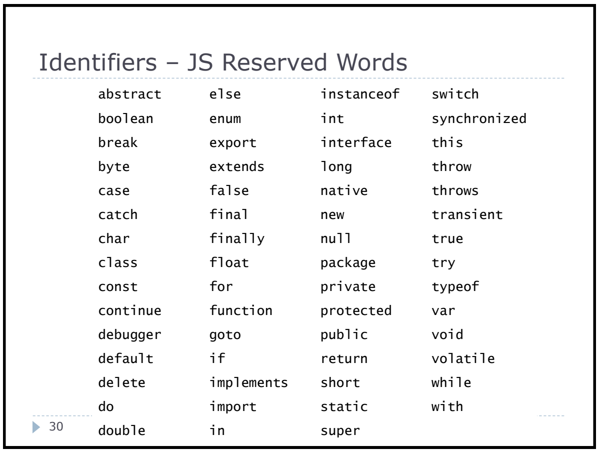
Example: `first_name`



Identifiers continued

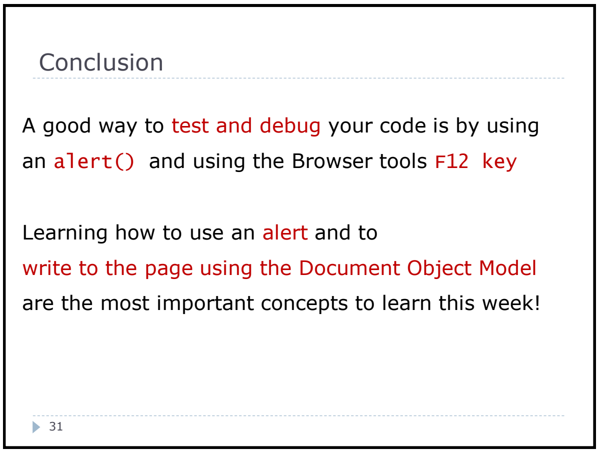
Rules for naming identifiers

- Begin with a letter, the $ character, or the underscore (_)
- Cannot begin with a number but can contain a number
- Cannot contain spaces, punctuation, mathematical or logical operators
- Cannot be JavaScript reserved words



Identifiers – JS Reserved Words

```
abstract
boolean
break
byte
case
catch
char
class
const
continue
debugger
default
delete
do
double
else
enum
export
extends
false
final
finally
float
for
function
goto
if
implements
import
in
instanceof
int
interface
long
native
new
null
package
private
protected
public
return
short
static
super
switch
synchronized
this
throw
throws
transient
true
try
typeof
var
void
volatile
while
with
```



Conclusion

A good way to test and debug your code is by using an `alert()` and using the Browser tools `F12 key`

Learning how to use an alert and to write to the page using the Document Object Model are the most important concepts to learn this week!