JavaScript

More Interactivity than CSS

JavaScript

Is: a programming language

- It is light-weight, interpreted or parsed, not compiled
- It is object-oriented
- It executes on the client

Not: Java

- It is based on ECMAScript, an open-source "universal" language
- Universally implemented in browsers

JavaScript

1. **Access Content**
   select any element, attribute, or text
2. **Modify Content**
   add or remove elements, attributes, and text to a page
3. **Program Rules**
   instructions for the browser
4. **React to Events**
   respond to the user or browser behavior

Document Object

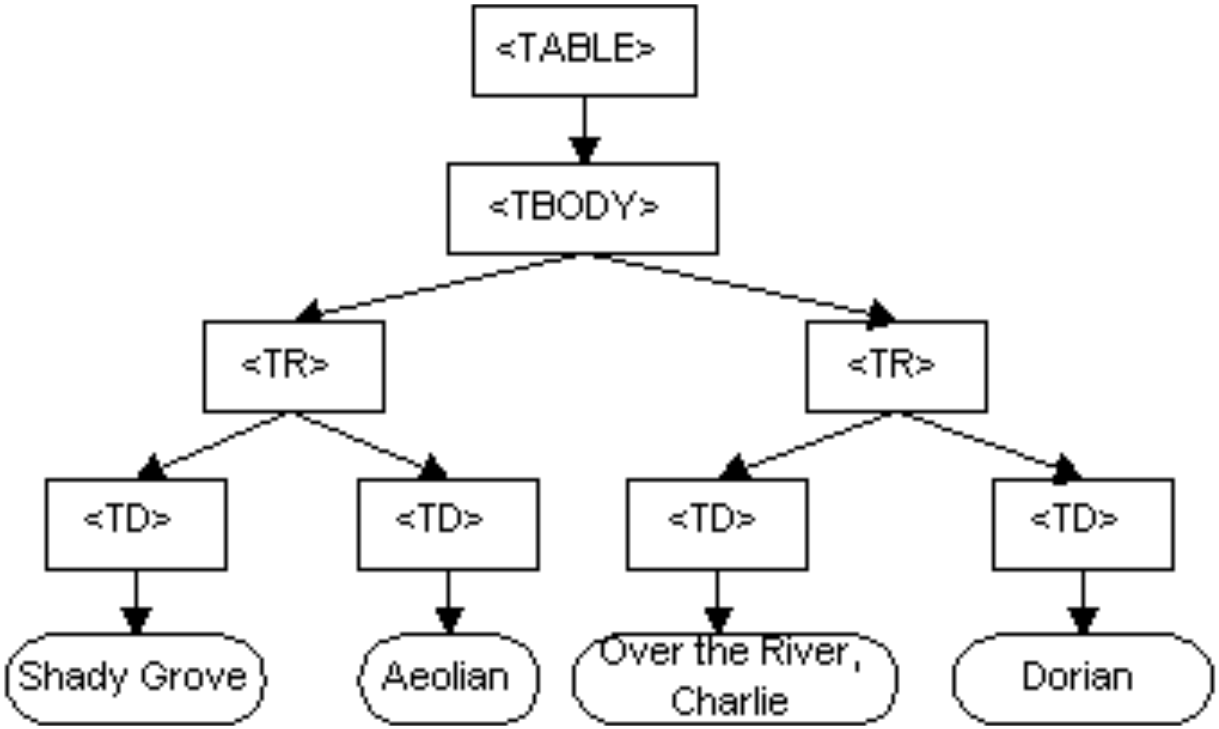
- An HTML document loaded into a web browser is a *document object*
- The document object is the *root* node and the owner of all other nodes
- The document object has *properties* and *methods* for accessing all node objects using JavaScript

Document Object

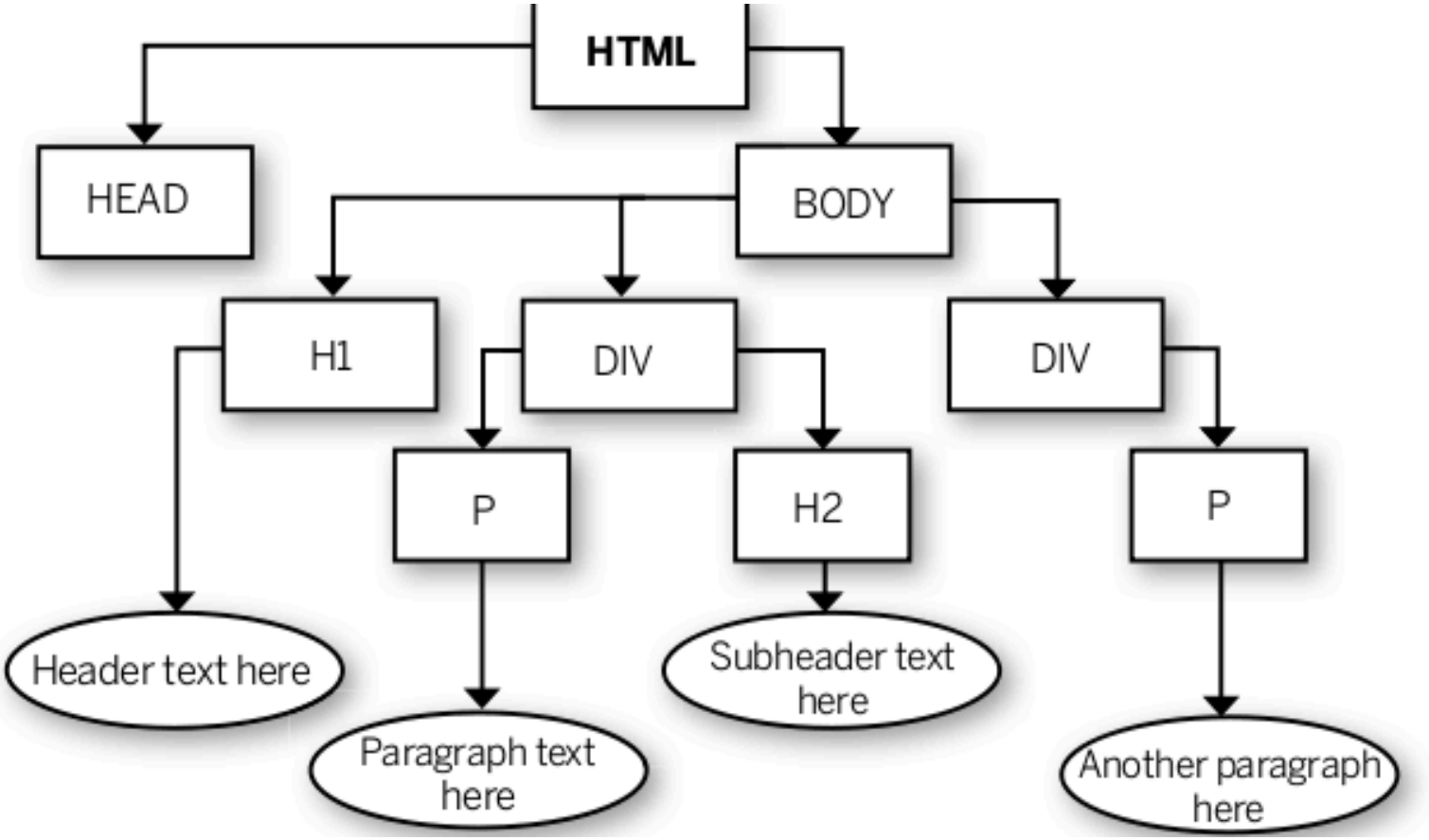
Everything is a node.

- The document is a document node
- All HTML elements are element nodes
- All HTML attributes are attribute nodes
- Even comments are comment nodes

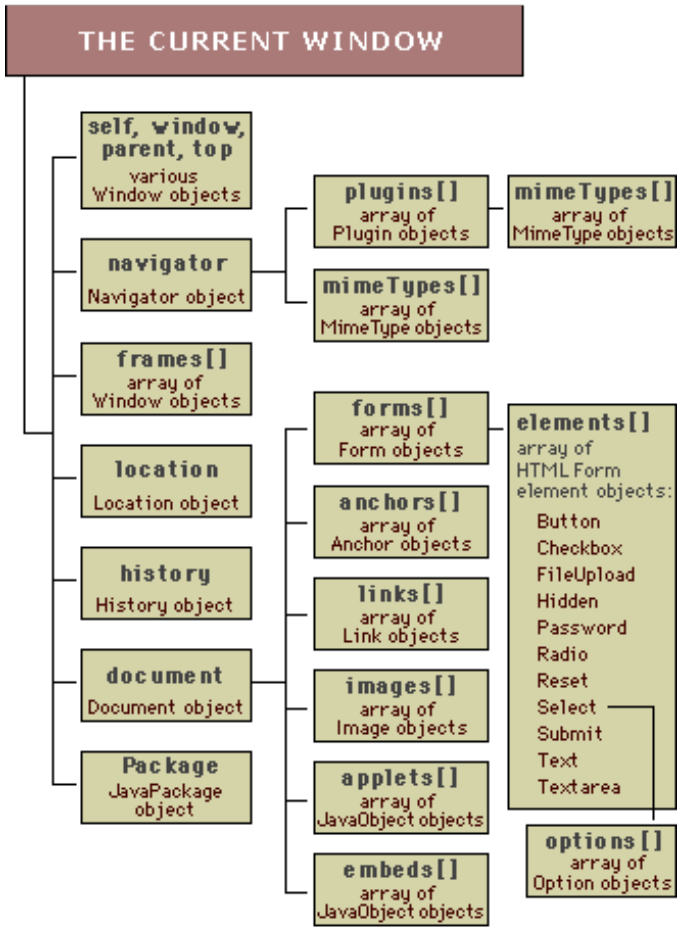
Document Object Example



Another DOM Example



Complex DOM Example



HTML DOM

The HTML DOM is a standard *object* model and *programming interface* for HTML

It describes how to *get*, *change*, *add*, or *delete* HTML elements

The DOM defines

- all HTML elements as objects
- properties of all HTML elements
- methods to access all HTML elements
- events for all HTML elements

Accessing HTML Content

Using the document object, we can use methods to find content.

```
document.querySelector(selectors);
parentNode.querySelectorAll(selectors);
```

```
document.getElementBy
                      Id
                      ClassName
                      TagName
```

Creating HTML Content

The following selects the <p> element with an id="demo", and changes the text content from "Hi everyone" to "Hello World!"

```
<!DOCTYPE html>
<html>
<body>
<h2>demo1</h2>

<p id="demo">Hi everyone</p>

<script>
document.getElementById("demo").innerText = "Hello
World!";
</script>

</body>
</html>
```

Demo1

Reacting to HTML Event

```
<!DOCTYPE html>
<html>
<head>
   <title>demo2</title>
</head>

<body>
    <h2>demo2</h2>
    <p id="demo">Hello World!</p>

    <button type="button"
        onclick="document.getElementById('demo').innerText =
Date()">
        Click me to display Date and Time.</button>

</body>
</html>
```

Any event will automatically invoke the JavaScript interpreter

Demo2

Changing HTML Styles (CSS)

```
<body>
<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change the style of an HTML element.</p>

<script>
    function myFunction() {
        var x = document.getElementById("demo");
        x.style.fontSize = "25px";
        x.style.color = "red";
    }
</script>

<button type="button" onclick="myFunction()">Click Me!</button>
</body>
```

Demo3

Changing HTML Attributes

```
<h1>demo4: JavaScript Can Change Images</h1>

<img id="myImage" onclick="changeImage()" src="pic_bulboff.gif">

<p>Click the light bulb to turn on/off the light.</p>

<script>
    function changeImage()
    {
        var x = document.getElementById('myImage');
        if (x.src.match("bulbon"))
            { x.src = "pic_bulboff.gif"; }
        else
            { x.src = "pic_bulbon.gif"; }
    }
</script>
```

Demo4

JavaScript Code Fundamentals

Comments

```
//This is a single line comment

/*This is a multiple
line comment*/
```

Development Environment

Firefox: Tools > Web Developer > Web Console

Chrome: Controls > More Tools > Developer Tools

Safari: Safari > Preferences, click Advanced,
then select
"Show Develop menu in menu bar."

Fundamentals

Code goes between a pair of <script> and </script> tags in body *or* head

- Older code may have a type attribute: **`<script type="text/javascript">`** but it is not required
- Code can be placed in an external file using the src attribute
  `<script src="myScript.js"></script>`

Fundamentals

Dot Notation

- Notation describes how to go down the DOM chain to find an element

All of these find the same element:

```
document.images.dan.src
document.images['dan'].src
document.images[2].src
document.dan.src
```

Programming Refresher

Note: The rest of this slide set makes the assumption the reader has had a course in some sort of programming language such as Java, Perl, Python, C, C++, C# (c sharp), etc.

However, this course does not have a programming language prerequisite so there will be no testing of the information or requirements that JavaScript will be included in project3.

Refresher (cont'd)

- Variables (type, scope, declaration)
- Decisions (if, switch)
- Loops (for, for...in, while, do...while)
- Functions

JS Objects

Everything is an object

- Window
- Document
- Navigator
- elements

Properties - Object Descriptions

- Name
- Length
- Source
- Value

Methods - Object Actions

- write()
- open()
- cloneNode()
- RemoveEventListener()

JS Events

Object 'Triggers'

- onload
- onunload
- onclick
- onmouseover
- onmouseout
- onmousemove
- onmousedown
- onmouseup
- onmove
- onresize
- onchange
- onsubmit
- onreset
- onresize
- onabort
- onblur
- onfocus
- ondblclick
- ondragdrop
- onerror
- onkeydown
- onkeypress
- onkeyup
- …and more!

JS Events

JavaScript Event vs. HTML attribute

- HTML attribute events are case **not** sensitive (**`onmouseover`** or **`onMouseOver`**)

  `<img src="pic.jpg" onmouseover="jsFunct();">` - OK!

  `<img src="pic.jpg" onMouseOver="jsFunct();">` - OK!

- JS event (written between <script> tags) MUST be lower case (**`onmouseover`**)

  `objName.onmouseover=functToHappen;` - OK!

  `objName.onMouseOver=functToHappen;` - Will Break!

- JS **is** case sensitive! (**`var x`** is a different than **var X**)

Functions

- Why functions?

  Convenient, cleaner code, re-usability, execute only when desired

- Function creation uses the function keyword with parentheses and curly braces { }

```
function myFunct(arg1, arg2){
    //code goes here!
}
```

- Functions are called by putting the name of the function in the code with an opening and closing parentheses, e.g.

```
myFunct();      or
myFunct(arg1, arg2);
```

Functions

- return can be used to
  - return a value, or
  - jump out of a function early
- Useful (pre-built) functions:
  - `setTimeout()`
  - `setInterval()`
  - `eval()`
  - `parseInt()`
  - `typeof()`
  - `Math.random()`
  - `Math.floor()`
  - `blur()`
  - `focus()`

Arrays

- Simple array construction: (all the following are same!)

```
var newArr = new Array(3);
newArr[0]="dan";
newArr[1]=34;
newArr[2]=6.1;

var newArr=new Array("dan",34,6.1);

var newArr = ["dan", 34, 6.1];
```

(If you thought everything in JS is an object, you could also say everything in JS is an array...)

Arrays

- Ordered set of indexed elements...
  - Ordinary vs. Associative
  - Indexed by numbers vs. name value pairs

    `document.images[0].width` - Index the array with a number

    `document.images['name'].width` - Index the array with a name

    `document.name.width` - simply a property of the document

So what's this all mean?

- JS is heavily used and is a major technology many in the industry are using to replace Flash, for example.
  - jQuery – a library of client-side JS routines
  - Angular 2 – a development platform (framework) for building mobile and desktop web applications
- You need to be able to:
  - recognize JS when you see it
  - incorporate it into your pages
  - adapt it to your purposes

In Class Exercise

- Find some (simple) JavaScript and incorporate it into your main class page.
  - Make it “yours” – change something in it so it was more than just copy-paste, even if it just means changing some text in the output
  - Include a paragraph of text describing where you got it from, how you changed it and what it does!

Fundamentals

Typing javascript: into the URL invokes this protocol, thus giving access to all of the JavaScript on this page!

- For example, typing

  ```
  javascript: alert("hello");
  ```

  will make the window execute the alert method

- Also, typing

  ```
  javascript: myFunction();
  ```

  will make whatever code is in **myFunction** execute!