Document Object Model (DOM)

INFO/CSE 100, Autumn 2004
Fluency in Information Technology

http://www.cs.washington.edu/100

What the heck is the DOM?

- Document Object Model
  » Your web browser builds a model of the web page (the document) that includes all the objects in the page (tags, text, etc)
  » All of the properties, methods, and events available to the web developer for manipulating and creating web pages are organized into objects
  » Those objects are accessible via scripting languages in modern web browsers

References

- JavaScript, The Definitive Guide
  - by David Flanagan. Publisher O'Reilly

- W3C Document Object Model
  - http://www.w3.org/DOM/
  - http://www.w3.org/2003/02/06-dom-support.html

- Document Object Model in Mozilla
  - http://www.mozilla.org/docs/dom/

This is what the browser reads (sampleDOM.html):

```html
<html>
<head>
  <title>Sample DOM Document</title>
</head>
<body>
  <h1>An HTML Document</h1>
  <p>This is a <i>simple</i> document.</p>
</body>
</html>
```

This is what the browser displays on screen.
Recall our simple GUI example

This GUI has several simple controls.

- Two buttons to control the results
- One text field to display the results
- One pair of radio buttons to control the display
- One button to reinitialize

setResults(resultString)

```javascript
<script type="text/javascript">
function setResults(resultString) {
  var tempString = resultString;
  if (document.getElementById("radioLC").checked) {
    tempString = tempString.toLowerCase();
  } else if (document.getElementById("radioUC").checked) {
    tempString = tempString.toUpperCase();
  }
  document.getElementById("resultField").value = tempString;
</script>
```

the highlighted script above makes reference to several objects in the document object model.

Why is this useful?

- Because we can access the model too!
  - the model is made available to scripts running in the browser, not just the browser itself
    - A script can find things out about the state of the page
    - A script can change things in response to events, including user requests
  - We have already used this capability in the GUI programming that we've done
- Reference to several nodes in the model of the page that the browser constructed

- `document`
  - The root of the tree is an object of type `HTMLDocument`
  - Using the global variable `document`, we can access all the nodes in the tree, as well as useful functions and other global information
    - title, referrer, domain, URL, body, images, links, forms, ...
    - open, write, close, `getElementById`, ...

- `getElementById("radioLC")`
  - This is a predefined function that makes use of the `id` that can be defined for any element in the page
  - An `id` must be unique in the page, so only one element is ever returned by this function
  - The argument to `getElementById` specifies which element is being requested

---

Some information from a document

```html
<html>
  <head>
    <title>DOM Sample A</title>
  </head>
  <body>
    Information about this document.
    <script type="text/javascript">
      document.write("Title: ", document.title);
      document.write("Referrer: ", document.referrer);
      document.write("Domain: ", document.domain);
      document.write("URL: ", document.URL);
    </script>
  </body>
</html>
```

Some information about elements

```html
<html>
  <head>
    <title>DOM Sample B</title>
    <script type="text/javascript">
      function showInfo() {
        var element = document.getElementById("opener");
        var buffer = element.id + " tag is " + element.tagName;
        alert(buffer);
        element = document.getElementById("actionItem");
        buffer = element.id + " tag is " + element.tagName;
        buffer += ", type is " + element.type;
        alert(buffer);
      }
    </script>
  </head>
  <body>
    <p id="opener">The id attribute is very helpful.</p>
    <p id="closer">This is the closing paragraph.</p>
    <form>
      <button id="actionItem" type="button" onclick="showInfo()">Show Info</button>
    </form>
  </body>
</html>
```
Some specific properties

```html
<head>
<title>Simple Sample GUI</title>
<script type="text/javascript">
function setResults(resultString) {
    var tempString = resultString;
    if (document.getElementById("radioLC").checked) {
        tempString = tempString.toLowerCase();
    } else if (document.getElementById("radioUC").checked) {
        tempString = tempString.toUpperCase();
    }
    document.getElementById("resultField").value = tempString;
}
</script>
</head>
```
Just the tip of the DOM

- The HTML Document Object Model is a standard for structuring data on a web page
  - The field is advancing rapidly as people recognize the benefits of standardized structure and access
  - The DOM is steadily improving to cover general purpose data structuring requirements
- XML (Extendible Markup Language) also uses the Core DOM to specify its structured data
  - similar to HTML but more carefully defined

This is what the browser reads (domC.html).

```html
<html>
<head>
<title>DOM Sample C</title>
<script type="text/javascript">
var switchCount = 0;
var adjectives = ["simple","complex","fascinating","unique"]; function switcher() {
  switchCount = (switchCount + 1) % adjectives.length;
  var italicNode = document.getElementById("adjPhrase");
  italicNode.firstChild.nodeValue = adjectives[switchCount];
}
</script>
</head>
<body>
<h1>An HTML Document</h1>
<p>This is a <i id="adjPhrase">simple</i> document. </p>
<form>
<button type="button" onclick="switcher()">switch</button>
</form>
</body>
</html>
```

This is what the browser displays on screen.