
Announcements

- Class Web Site:
 - <http://www.cs.umd.edu/projects/passport/Classes/Spring2008/>
 - You can find this link at the end of the main passport site
 - <http://www.cs.umd.edu/projects/passport/webPage/>
- Envelops!
- Certificates
- Project #4

Computer Science

- What is Computer Science?
- What areas can we identify in computer science?
- Is a programmer a computer scientist?
- How can you become a computer science major?

One-Dimensional Arrays

- Let's review and see some additional information about arrays.
- **Array** – ordered collection of values.
- **Indexing** – first element associated with index 0.
- An element of an array can be of any type and an array can hold different types of elements.
- **Initialization of arrays**
 - Via array literal – comma separated list of elements within square brackets.
 - `var a = [2, 3, 5];`
 - `var b = []; // empty array`
 - Specified in the Array constructor
 - `var c = new Array();`
 - `var d = new Array(2, 3, 5); // initializes array with 2, 3, 5`
 - `var e = new Array(4); // defines array of size 4`
- You can print array contents with alert.

One-Dimensional Arrays

- You can change the number of elements of an array at any time.

- Example

```
var data=[]; // initially zero elements
data[0]=10;
data[1]=20; // two elements by now
```

- **length property**

- Read/write value.
- Can be used to expand/truncate array.
- Example (truncating)

```
var data=[];
data[0]=10;
data[1]=20;
data.length = 1;
```

- **Example:** [GetFilteredData.html](#)

Converting Between Arrays and Strings

- **From array to String via join:**

```
var a = [5, 9, 10];  
var aStr = a1.join();           // aStr → 5, 9, 10  
var aStr2 = a1.join("<br />"); // aStr2 → 5<br />9<br />10<br />
```

- **From String to array via split:**

```
var b = "30, 40, 50";  
var bArray = b.split(",");  
for (var idx=0; idx < bArray.length; idx++) // loop that prints 30, 40, 50  
    alert(bArray[idx]);
```

Combining and Dividing Arrays

- **concat** – join arrays together returning an array with the result. It does not modify the original arrays. You can pass additional array as comma-delimited parameters.

- **Example:**

```
var data1 = ["Mary", "June"];  
var data2 = ["Lynn", "Kim", "Tim"];  
var concatenated = data1.concat(data2); // concatenated we will have  
// ["Mary", "June", "Lynn", "Kim", "Tim"]
```

- **splice** – removes a segment of elements from the array. First parameter represents starting index; second number of elements to retrieve from that index on.

- **Example:**

```
var ages = [-1, 4, 5, 6, 8, 10, 12, 14];  
var segment = ages.splice(2, 3);  
// ages now has the values -1,4,10,12,14  
// segment has the values 5,6,8
```

Two-Dimensional Arrays

- JavaScript does not support actual two-dimensional arrays
- You can simulate two-dimensional arrays by using arrays of arrays.
- About two-dimensional arrays
 - You can pass them and return them from functions like one-dimensional arrays.
 - Any modifications in the function will be permanent.
 - You can have ragged arrays.
- **Example:** `TwoDimensionalArrays.html`

Functions as Data

- In JavaScript functions are considered data.
- That means they can be assigned to variables, stored as properties of objects or elements of arrays, passed as arguments to functions, etc.
- **Example:** FunctionsAreData.html
- Where have we seen this?

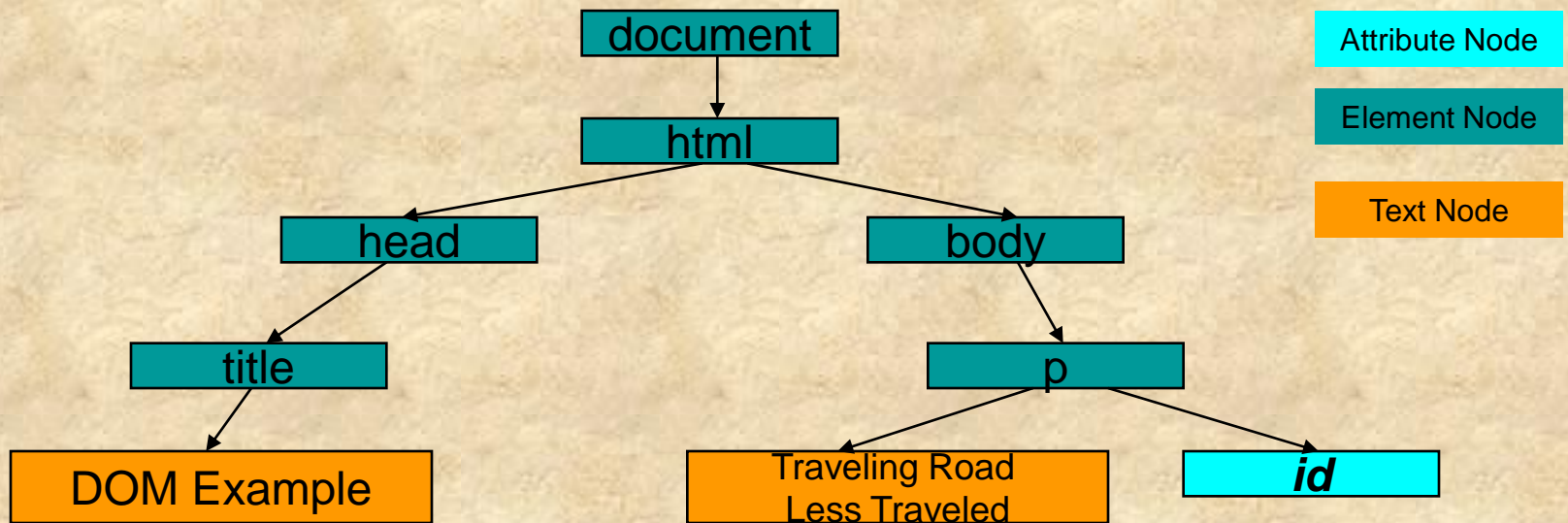
DOM (Document Object Model)

- **DOM** – representation of the elements of a web page (e.g., headings, lists, paragraphs, styles, etc.) used by a JavaScript programs to manipulate web page elements.
- **DOM** – Allows JavaScript programs to **dynamically** access and update the content, structure, and style of documents.
 - From a JavaScript program you can control the image displayed in your page every hour.
 - From a JavaScript program you can let users decide what background color to use.
 - You could add/remove new items from a list.
 - Others.

DOM (Document Object Model)

- DOM represents elements of a web page as a tree structure consisting of nodes.
 - Each pair of tags (.e.g, <p>, </p>) is represented by a node.
 - Three types of nodes:
 - text nodes
 - element nodes
 - attribute nodes
- Manipulation of these node allows a JavaScript program to access any information present in a web page.

Example DOM for HTML File



```
<html>
  <head><title>DOM Example</title></head>
  <body>
    <p id="message">Traveling the road less traveled. </p>
  </body>
</html>
```

DOM (Document Object Model)

- To access any element of your web page you could traverse the tree.
 - Easier approach:
 - **document.getElementById** method
 - Returns element with specified id
 - **getElementsByTagName** method
 - Can be used with document and every single element node
 - Returns a list of nodes (array)
- To read and write attributes of an element “elem”
 - `elem.getAttribute(nameOfAttribute)`
 - `elem.setAttribute(nameOfAttribute, newValue)`
- **Example:** AccessingDom.html

Animations

- We can create animations by using the DOM
- **Example:** `AccessingDOMII.html`

DHTML

- **DHTML (Dynamic HTML)** – It is a combination of HTML, CSS, and JavaScript where scripts dynamically alter the style of a document.

Form Validation

- Forms – We can validate the data associated with a form by recognizing the submit event.
- Keep in mind that JavaScript can be disabled therefore always validate data on the server side.
- **Example:** FormValidation.html
 - Illustrates fieldset/legend.
 - Illustrates alternatives to access form values.
 - elements array
 - document.getElementById

Debugger

- Allow us to see values of variables.
- Allow us to execute statements one at a time.
- Firebug Debugger.
 - <http://www.getfirebug.com/>
- Demo