Document Object Model (DO

Junghoo Cho

cho@cs.ucla.edu

HTML DOM (Document Object Model)

- Standard to construct “objects” from an HTML document
- HTML document is converted to a tree-like model
  - “DOM tree”
- JavaScript manipulates elements on a Web page through DC

Adding JavaScript to a Web page

- Use `<script>` tag
  - Direct embedding

    ```
    <script>
        ... javascript code ...
    </script>
    ```

  - Linking to a separate file

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

- `<script>` may appear anywhere on a page

Document to DOM Tree

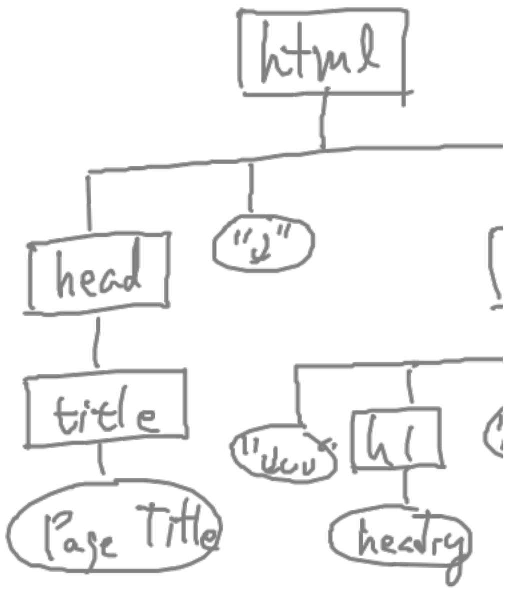
Three key node types

1. *Element node*: An HTML element
   - Every HTML tag creates an element node
2. *Text node*: Text enclosed in an element
   - Text node becomes a child of the element node
3. *Attribute node*: Attribute of an element
   - An attribute node is *associated with* the element node, but is not a c

DOM Conversion Example

```
<!DOCTYPE html>
<html>
<head><title>Page Title</title></head>
<body>
    <h1>Heading</h1>
    <a href="good/">Link</a>
</body>
</html>
```

- White spaces are preserved from `<head>` through `</body>`



DOM Tree in JavaScript

- Every DOM node becomes a JavaScript object
  - with *properties*, *methods*, and associated *events*
- `document` object
  - "root" object that has the parsed DOM tree as its "child"
- DOM object properties for tree structure
  - `childNodes`: the node's children
  - `parentNode`: the node's parent
  - `attributes`: the node's attribute

DOM Object Properties

- `nodeType`: Node type
  - 1: Element, 2: Attribute, 3: Text, …
- `nodeName`
  - Tag name for element node (e.g., `HEAD`)
  - Attribute names for and attribute node (e.g. `href`)
  - `#text` for text node, …
- `nodeValue`
  - Enclosed text for text and comment nodes
  - Attribute value for attribute nodes
  - `null` otherwise

Accessing DOM Node

1. Traverse the tree using the `childNodes` property starting fro `document`
2. Get node(s) directly

```
document.getElementByID('id');
document.getElementsByTagName('h1');
document.getElementsByClassName('class');
document.querySelectorAll('body > a')
```

- DOM traversal using Chrome Developer Console

Manipulating DOM Nodes

- JavaScript objects corresponding to DOM nodes have
  - Properties
  - Methods
  - Associated events
- By changing the property values, calling the methods, we ca the HTML element dynamically

DOM Manipulation Example (1)

```
document.body.style.background = "yellow";
document.getElementById('warning1').style.color = "red";
document.body.innerHTML = "<p>new text</p>";
```

- `innerHTML` value is parsed into a “DOM tree” and replaces th child(ren)

DOM Manipulation Example (2)

```
let newP = document.createElement("p");
let newText = document.createTextNode("new text");
newP.appendChild(newText);
document.body.replaceChild(newP);
```

- `document.createElement()`, `createTextNode()`, `appendCh` `removeChild()`, and `replaceChild()` can be used to add an DOM objects

DOM Manipulation Example (3)

```
document.getElementById('myform1').reset();
document.getElementById('myform1').submit();
```

- Object method can be called to take a certain action

Event-Driven Programming

- To dynamically update a Web page based on user action, Jav
  program must
  1. "Wait for" relevant "events"
  2. Take appropriate actions given an event

Event Handling in JavaScript

- Every DOM object is associated with a set of “events”
  - e.g., `load`, `click`, `input`, `mouseover`, …
- An object has an associated *event handler* for each event
  - A function to be *invoked* when the event is *triggered*
  - We can customize its action by setting the event handler
- When an event is fired (= *triggered*) on an object (= *event targ* associated callback function (= *event handler*) is called (= *inv*

Setting Event Handler

1. In JavaScript: `obj.addEventListener(event, handler)`

```
function ChangeColor(event) {
    document.body.style.color = "red";
}
document.body.addEventListener("click", ChangeColor);
```

2. In HTML: `onevent="stmt;"` attribute:

```
<body onclick="ChangeColor();">
```

   - Not recommended

Our Demo Code

```
<html>
<meta charset="utf-8">
<head><title>JavaScript Example</title></head>
<body>Click on this document!</body>
<script>
    let colors = [ "yellow", "blue", "red" ];
    let i=0;
    function ChangeColor(event) {
        document.body.style.backgroundColor = colors[i++%3];
    }
    document.body.addEventListener("click", ChangeColor);
</script>
</html>
```

References

- W3C DOM
- W3C DOM Events
- More accessible reference for common JavaScript and DOM