

JavaScript Browsers Objects

CSC 342 - Web Technologies

JavaScript Object Types

- User defined objects
- Native objects (Array, Math, Date, etc.)
- Host Objects provided by the browser
 - The window object is a representation of the browser
 - The document object is a representation of the contents of the web page; it is a property of the window object

The Document Object Model (DOM)

- The HTML Document Object Model (DOM) represents the HTML document as a tree of *node* objects
- `window.document` is the root node of the HTML document
- Types of nodes:
 - The document is a *document* node
 - HTML elements are *element* nodes
 - HTML attributes are *attribute* nodes
 - Text (content) inside HTML elements are *text* nodes
 - Comments are *comment* nodes

Traversing Nodes in the DOM Tree

- Node properties
 - parentNode
 - childNodes
 - firstChild
 - lastChild
 - nextSibling
 - previousSibling

Searching for Nodes in the DOM Tree

- document methods

- `getElementById(id)`
- `getElementsByTagName(name)`
- `getElementByClassName(name)`
- `querySelector(CSS selector)`
- `querySelectorAll(CSS selector)`

- document properties

- `anchors`
- `forms`
- `images`
- `links`
- `scripts`

NodeLists and HTMLCollections

- Document methods, such as `document.getElementsByName`, return a `NodeList` object
- Document properties, such as `document.forms`, are `HTMLCollection` objects
- `NodeList` and `HTMLCollection` objects are read-only array-like objects
- `NodeList` and `HTMLCollection` objects are *live* – the list of elements that they contain change as the document changes

Manipulating the DOM Tree

- `document.createElement(element)`
- `element.removeChild(element)`
- `element.appendChild(element)`
- `element.replaceChild(new element, old element)`
- `element.setAttribute(attribute, value)`
- `element.style.property = new style value`
Note: CSS properties use the camel case naming convention in JavaScript if they include hyphens, for example `font-size` is named `fontSize`

HTML DOM Events

- Browser based JavaScript programs use an event-driven programming model.
- The web browser generates an *event* when something interesting happens to the document or browser:
- An HTML event is a *thing* that happens to an HTML element
- Examples of HTML events:
 - The web page is loaded
 - The user clicks the mouse
 - The mouse moves over an element
 - A key is pressed
 - An input field is changed
 - A form is submitted

Handling DOM Events

- An *event handler* or *event listener* is a JavaScript function that is executed when an HTML event occurs
- Registering an event listener to an HTML element

- HTML event attributes

```
<button onclick="someFunction()">
```

- HTML DOM node property

```
<script>  
document.getElementById("button").onclick  
    = someFunction;  
</script>
```

- The `addEventListener` method (recommended)

```
<script>  
document.getElementById("button")  
    .addEventListener("click", someFunction);  
</script>
```

The addEventListener Method

`element.addEventListener(event, function, useCapture)`

- Attaches an event handler without overwriting existing event handlers
- Can add multiple event handlers to one element (can be of the same type)
- Can add an event handler to any DOM object, for example, the `window` object
- Can remove specific event handlers with the `removeEventListener` function
- Can control event bubbling

JavaScript Event Object

- A JavaScript Event object is created when an event occurs
- A JavaScript event handler can access the Event object by passing it in as a parameter

```
<script>  
function handleEvent(e) {  
    console.log(e.target);  
}  
</script>
```

- There are different event objects for different events, for example, the `MouseEvent` object

The Browser Object Model (BOM)

- The BOM is not a standard – each browser has a different implementation
- Interesting properties of the window object:
 - `location`: object representing the current URL
 - `history`: object representing URLs previously visited
 - `navigator`: object representing the browser
- Interesting methods of the window object:
 - `setTimeout()`
 - `setInterval()`