CNU 국립 충남대학교 CHUNGNAM NATIONAL UNIVERSITY

9. JS HTML DOM & Browser BOM

웹프로그래밍

2016년 1학기

충남대학교 컴퓨터공학과

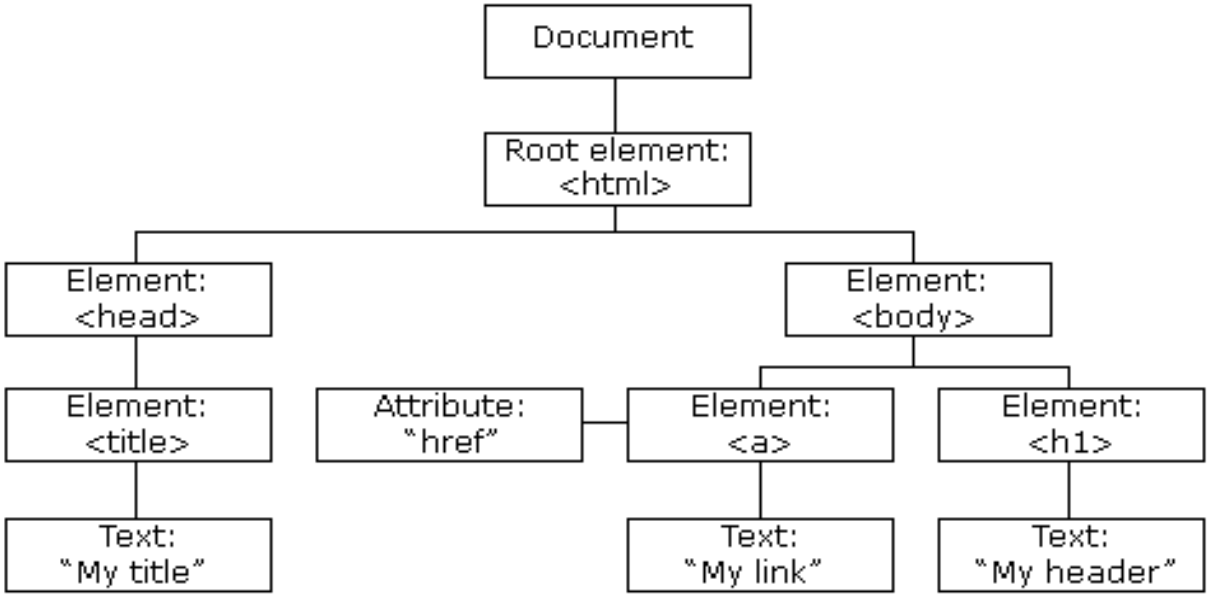
목차

- **DOM (Document Object Model)**
  - DOM Introduction
  - DOM Methods
  - DOM Document
  - DOM Events
  - DOM Nodes
- **BOM (Browser Object Model)**
  - BOM Display
  - BOM Location
  - BOM History
  - BOM Navigator
  - BOM Popup
  - BOM Timing
  - BOM Cookies

JavaScript HTML DOM

- **The HTML DOM (Document Object Model)**
  - When a web page is loaded, the browser creates a Document Object Model of the page.
  - The HTML DOM Tree of Objects

```
<html>
  <head>
    <title> My title</title>
  </head>
  <body>
     <a  href="" > My Link</a>
     <h1> My header</h1>
  </body>
</html>
```



JavaScript HTML DOM (cont'd)

- **With the object model, JavaScript gets all the power it needs to create dynamic HTML:**
  - JavaScript can change all the HTML elements in the page.
  - JavaScript can change all the HTML attributes in the page.
  - JavaScript can change all the CSS styles in the page.
  - JavaScript can remove existing HTML elements and attributes.
  - JavaScript can add new HTML elements and attributes
  - JavaScript can react to all existing HTML events in the page.
  - JavaScript can create new HTML events in the page.

JavaScript HTML DOM (cont'd)

- **What is the DOM?**
  - A W3C (World Wide Web Consortium) standard
  - Defines a standard for accessing documents
    - *"The W3C Document Object Model(DOM) is platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."*
  - Separated into 3 different parts
    - Core DOM – standard model for all document types
    - XML DOM – standard model for XML documents
    - HTML DOM – standard model for HTML documents

JavaScript HTML DOM (cont'd)

- **What is the HTML DOM?**
  - A standard **object** model and **programming interface** for HTML. It defines:
    - The HTML elements as **objects**
    - The **properties** of all HTML elements
    - The **methods** to access all HTML elements
    - The **events** for all HTML elements

JavaScript – HTML DOM Methods

- **The DOM Programming Interface** *Try it!*
  - The HTML DOM can be accessed with JavaScript (and with other programming languages).
  - In the DOM, all HTML elements are defined as **objects**.
  - The programming interface is the properties and methods of each object.
    - A **property** is a value that you can get or set (like changing the content of an HTML element). (e.g., innerHTML)
    - A **method** is an action you can do (like add or deleting an HTML element). (e.g., getElementById)

JavaScript HTML DOM Document

- **The HTML DOM Document Object**
  - The document object represents your web page.
  - The document object is the owner of all other objects in your web page.
  - If you want to access objects in an HTML page, you always start with accessing the document object.

- **Finding HTML Elements**

Method	Description
document.getElementById()	Find an element by element id
document.getElementsByTagName()	Find elements by tag name
document.getElementsByClassName()	Find elements by class name

JavaScript HTML DOM Document (cont'd)

Changing HTML Elements

Method	Description
element.innerHTML=	Change the inner HTML of an element
element.attribute=	Change the attribute of an HTML element
element.setAttribute(*attribute,value*)	Change the attribute of an HTML element
element.style.*property*=	Change the style of an HTML element

Adding and Deleting Elements

Method	Description
document.createElement()	Create an HTML element
document.removeChild()	Remove an HTML element
document.appendChild()	Add an HTML element
document.replaceChild()	Replace an HTML element
document.write(*text*)	Write into the HTML output stream

JavaScript HTML DOM Document (cont'd)

- **Adding Events Handlers**

Method	Description
document.getElementById(*id*).onclick=function() {*code*}	Adding event handler code to an onclick event

- **Finding HTML Objects** ***Refer to it!***
  - The first HTML DOM Level 1 (1998), defined 11 HTML objects, object collections, and properties. These are still valid in HTML5.
  - Later, in HTML DOM Level 3, more objects, collections, and properties were added.

JavaScript HTML DOM Navigation

- **You can navigate the node tree using node relationships.**
- **DOM Nodes**
  - According to the W3C HTML DOM standard, everything in an HTML document is a node:
    - The entire document is a document node
    - Every HTML element is an element node
    - The text inside HTML elements are text nodes
    - Every HTML attribute is an attribute node
    - All comments are comment nodes
  - With the HTML DOM, all nodes in the node tree can be accessed by JavaScript.
  - New nodes can be created, and all nodes can be modified or deleted.

JavaScript HTML DOM Navigation (cont'd)

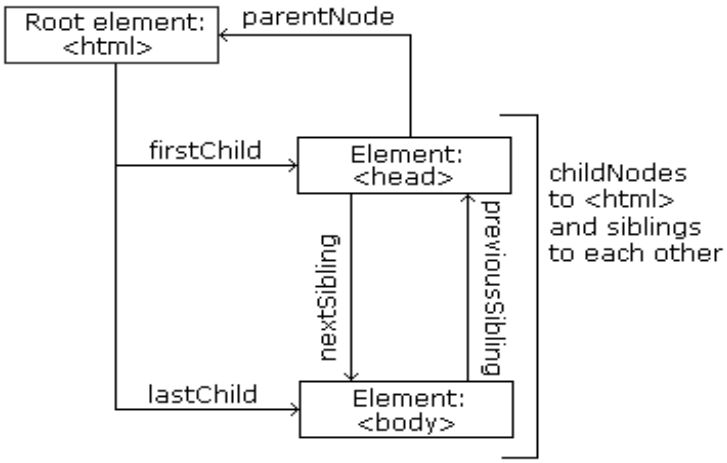
Node Relationships

- The nodes in the node tree have a hierarchical relationship to each other.
- The terms parent, child, and sibling are used to describe the relationships.
  - In a node tree, the top node is called the root (or root node).
  - Every node has exactly one parent, except the root (which has no parent).
  - A node can have a number of children.
  - Siblings are nodes with the same parent.

```
<html>
  <head>
    <title> DOM Tutorial</title>
  </head>

  <body>
    <h1> DOM Lesson one</h1>
    <p>Hello world!</p>
  </body>

</html>
```



JavaScript HTML DOM Navigation (cont'd)

- **Navigating Between Nodes**
  - Node properties to navigate between nodes with JavaScript:
    - parentNode , childNodes[*nodenumber*] , firstChild, lastChild, nextSibling, previousSibling
- **Child Nodes and Node Values** *Try it!* *Try it!*
- **DOM Root Nodes**
  - Two special properties that allow access to the full document.
    - document.body - the body of the document *Try it!*
    - document.documentElement - the full document *Try it!*

JavaScript HTML DOM Navigation (cont'd)

- **The nodeName property**
  - Specifies the name of a node.
    - nodeName is read-only.
    - nodeName of an element node is the same as the tag name.
    - nodeName of an attribute node is the attribute name.
    - nodeName of a text node is always #text.
    - nodeName of the document node is always #document.
  - **Note:** nodeName always contains the uppercase tag name of an HTML element.
- **The nodeValue property**
  - Specifies the value of a node.
    - nodeValue for element nodes is undefined
    - nodeValue for test nodes is the text itself
    - nodeValue for attribute nodes is the attribute value

JavaScript HTML DOM Navigation (cont'd)

- **The nodeType property**
  - Returns the type of node. nodeType is read-only.
  - The most important node types are:

Element type	NodeType
Element	1
Attribute	2
Text	3
Comment	8
Document	9

JavaScript HTML DOM Elements (Nodes)

- **Adding and Removing Nodes (HTML Elements)**
- **Creating New HTML Elements (Nodes)** *Try it!*
  1. Create the element (element node).
  2. Append it to an existing element.
  - **appendChild( )** method appended the new element as the last child of the parent.
- **Creating New HTML Elements – insertBefore( )** *Try it!*
- **Removing Existing HTML Elements** *Try it!*
- **Replacing HTML Elements** *Try it!*

JavaScript HTML DOM Node List

- **A node list is a collection of nodes.**
- **HTML DOM Node List** *Try it!*
  - **getElementsByTagName( )** method returns a **node list.**
  - A node list is an array-like collection of nodes.
    - The index starts at 0.
- **HTML DOM Node List Length** *Try it!* *Try it!*
  - The length property defines the number of nodes in a node-list:

```
var myNodelist = document.getElementsByTagName("p");
document.getElementById("demo").innerHTML = myNodelist.length;
```

JavaScript HTML DOM Elements

- **Finding HTML Elements**
  - By id, tag name, class name, CSS selectors, HTML object collections.
- **Finding HTML Element by Id** *Try it!*
- **Finding HTML Elements by Tag Name** *Try it!* *Try it!*
- **Finding HTML Elements by Class Name** *Try it!*
- **Finding HTML Elements by CSS Selectors** *Try it!*
- **Finding HTML Elements by HTML Object Collections** *Try it!*

JavaScript HTML DOM – Changing HTML

- **Changing the HTML Output Stream** *Try it!*
  - document.write( ) can be used to write directly to the HTML output stream.
- **Changing HTML Content** *Try it!* *Try it!*
  - Using the **innerHTML** property

```
document.getElementById(id).innerHTML = new HTML
```

- **Changing the Value of an Attribute** *Try it!*

```
document.getElementById(id).attribute = new value
```

- ***Test Yourself with Exercises!***

JavaScript HTML DOM – Changing CSS

- **Changing HTML Style** *Try it!*

  document.getElementById(*id*).style.*property* = *new style*

- **Using Events** *Try it!*
  - The HTML DOM allows you to execute code when an event occurs.
  - Events are generated by the browser when "things happen" to HTML elements:
    - An element is clicked on.
    - The page has loaded.
    - Input fields are changed.

- *HTML DOM Style Object Referenece*
- *Test Yourself with Exercises!*

JavaScript HTML DOM Animations

- **A Basic Web Page** *Try it!*
  - To demonstrate how to create HTML animations with JavaScript
- **Create an Animation Container**
  - All animations should be relative to a container element.

```
<div id ="container">
    <div id ="animate">My animation will go here</div>
</div>
```

- **Style the Elements** *Try it!*
  - The container element should be created with style = "position: relative".
  - The animation element should be created with style = "position: absolute".

JavaScript HTML DOM Animations (cont'd)

- **Animation Code**
  - JavaScript animations are done by programming gradual changes in an element's style.
  - The changes are called by a timer. When the timer interval is small, the animation looks continuous. The basic code is:

```
var id = setInterval(frame, 5);

function frame() {
    if (/* test for finished */) {
        clearInterval(id);
    } else {
        /* code to change the element style */
    }
}
```

- **Create the Animation Using JavaScript** *Try it!*

JavaScript HTML DOM Events *Try it!*

- **Reacting to Events**
  - To execute code when a user clicks on an element, add JavaScript code to an HTML event attribute.

    onclick = *JavaScript*

  - Examples of HTML events:
    - When a user clicks the mouse *Try it!* *Try it!*
    - When a web page has loaded
    - When an image has been loaded
    - When the mouse moves over an element
    - When an input field is changed
    - When an HTML form is submitted
    - When a user strokes a key

JavaScript HTML DOM Events (cont'd)

- **Assign Events Using the HTML DOM** *Try it!*
- **The onload and onunload Events** *Try it!*
  - Are triggered when the user enters or leaves the page.
- **The onchange Event** *Try it!*
- **The onmouseover and onmouseout Events** *Try it!*
- **The onmousedown, onmouseup and onclick Events** *Try it!*

JavaScript HTML DOM Events (cont'd)

- **More Examples**
  - Change an image when a user holds down the mouse button. *Try it!*
  - Display an alert box when the page has finished loading. *Try it!*
  - Change the background-color of an input field when it gets focus. *Try it!*
  - Change the color of an element when the cursor moves over it. *Try it!*

- HTML DOM Event Object Refernce

- *Test Yourself with Exercises!*

JavaScript HTML DOM EventListener

- **The addEventListener( ) method** *Try it!*
  - Syntax

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

    - First parameter is the type of the event (like "click", or "mousedown").
    - Second parameter is the function we want to call when the event occurs.
    - Third parameter is a Boolean value specifying whether to use event bubbling or event capturing. (optional)
- **Add an Event Handler to an Element** *Try it!* *Try it!*
- **Add Many Event Handlers to the Same Element** *Try it!*

JavaScript HTML DOM EventListener (cont'd)

- **Add an Event Handlers to the Window Object** *Try it!*
- **Passing Parameters :** "anonymous function" *Try it!*
- **Event Bubbling or Event Capturing** Try it!
- **The removeEventListener( ) method** *Try it!*
- HTML DOM Event Object Reference

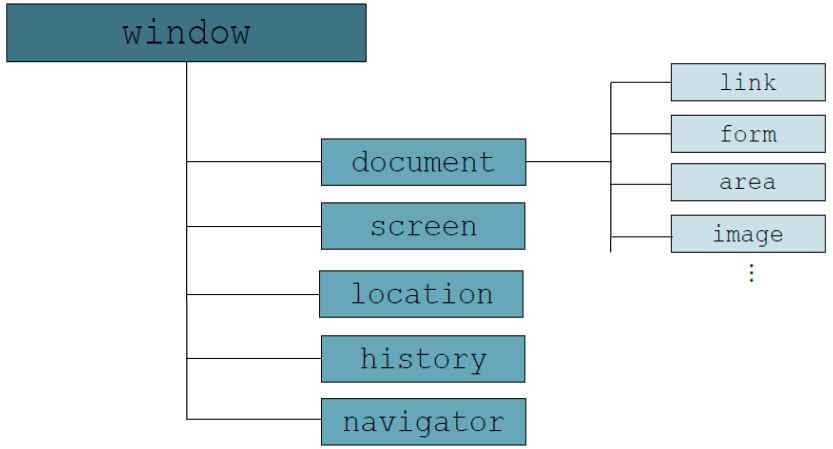
JavaScript Window – The Browser Object Model

- **Brower Object Model (BOM) allows JavaScript to "talk to" the browser.**

- **The Browser Object Model**
  - There are no official standards for the **B**rowser **O**bject **M**odel.
  - Since modern browsers have implemented (almost) the same methods and properties for JavaScript interactivity, it is often referred to, as methods and properties of the BOM.

JavaScript Window – The Browser Object Model (cont'd)

Object Name	Represents
window	The browser window
document	The current web page
screen	The screen area occupied by the browser and page
location	The URL of the current web page
history	The list of pages the user has visited previously
navigator	The web browser you're using



JavaScript Window – The Browser Object Model (cont'd)

- **The Window Object**
  - Represents the browser's window.
  - All global JavaScript objects, functions, and variables automatically become members of the window object.
    - Global variables are properties of the window object.
    - Global functions are methods of the window object.
    - Document object (of the HTML DOM) is a property of the window object.

```
window.document.getElementById("header");
```

is the same as

```
document.getElementById("header");
```

JavaScript Window – The Browser Object Model (cont'd)

- **Window Size** *Try it!*
  - Can used to determine the size of the browser window (the browser viewport, Not including toolbars and scrollbars).
  - **window.innerHeight** – the inner height of the browser window
  - **window.innerWidth** – the inner width of the browser window
- **Other Window Methods**
  - window.open( ) – open a new window
  - window.close( ) – close the current window
  - window.moveTo( ) – move the current window
  - window.resizeTo( ) – resize the current window

JavaScript Window Screen

- **The window.screen object contains information about the user's screen.**
- **Window Screen**
  - The **window.screen** object can be written without the window prefix.
  - Properties
    - screen.width *Try it!*
    - screen.height *Try it!*
    - screen.availWidth *Try it!*
    - screen.availHeight *Try it!*
    - screen.colorDepth *Try it!*
    - screen.pixelDepth *Try it!*

JavaScript Window Location

- **The window.location object can be used to get the current page address(URL) and to redirect the browser to a new page.**

- **Window Location**
  - window.location object can be written without the window prefix.
  - window.location.href *Try it!*
  - window.location.hostname *Try it!*
  - window.location.pathname *Try it!*
  - window.location.protocol *Try it!*
  - window.location.assign *Try it!*

JavaScript Window History

- **The window.history object contains the browsers history.**
- **Window History** *Try it!*
  - history.back( )
    - Same as clicking back in the browser
  - history.forward( )
    - Same as clicking forward in the browser

JavaScript Window Navigator

- **The window.navigator object contains information about the visitor's browser.**

- **Window Navigator**
  - navigator.appName
  - navigator.appCodeName *Try it!*
  - navigator.platform *Try it!*

IE11, Chrome, Firefox, and Safari return appName "Netscape".

Chrome, Firefox, IE, Safari, and Opera all return appCodeName "Mozilla".

JavaScript Window Navigator (cont'd)

- **Navigator Cookie Enabled** *Try it!*
  - The property cookieEnabled returns true if cookies are enabled.
- **The Browser Engine** *Try it!*
  - The properties product returns the engine name of the browser.
- **The Browser Language** *Try it!*
  - The property **language** returns the browser's language.
- **Is Java Enabled?** Try it!
  - The method **javaEnabled()** returns true if Java is enabled.

JavaScript Window Navigator (cont'd)

- **The Browser Version I** *Try it!*
  - The property appVersion returns version information about the browser.
- **The Browser Version II** *Try it!*
  - The property userAgent also returns version information about the browser.
- **Warning!!!**
  - Different browsers can use the same name.
  - The navigator data can be changed by the browser owner.
  - Some browsers misidentify themselves to bypass site tests.
  - Browsers cannot report new operating systems, released later than the browser.

JavaScript Popup Boxes

- **Alert Box** *Try it!*
  - Often used if you want to make sure information comes through to the user.
  - When an alert box pops up, the user will have to click "OK" to proceed.
  - Syntax:

```
window.alert("sometext");
```

- **Confirm Box** *Try it !*
  - Often used if you want the user to verify or accept something.
  - If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.
  - Syntax:

```
window.confirm("sometext");
```

JavaScript Popup Boxes (cont'd)

- **Prompt Box** *Try it!*
  - Often used if you want the user to input a value before entering a page.
  - If the user clicks "OK", the box returns the input value. If the user clicks "Cancel", the box returns null.
  - Syntax:

```
window.prompt("sometext", "defaultText");
```

- **Line Breaks** Try it!
  - To display line breaks inside a popup box, use a back-slash followed by the character n.

JavaScript Timing Events

- **JavaScript Timing Events**
  - Timing events: to execute some code at specified time-intervals.
- **The setInterval( ) Method** *Try it!* *Try it! (just like a digital watch)*
  - Executes a function, over and over again, at specified time intervals.
  - Syntax:

```
window.setInterval("javascript function", milliseconds);
```

- **How to Stop the Execution?** *Try it!*
  - Syntax:

```
window.clearInterval (intervalVariable);
```

JavaScript Timing Events (cont'd)

- **The setTimeout( ) Method** *Try it!*
  - Executes a function, once, after waiting a specified number of milliseconds.
  - Syntax:

```
window.setTimeout("javascript function", milliseconds);
```

- **How to Stop the Execution?** *Try it!*
  - Syntax:

```
window.clearTimeout (intervalVariable);
```

JavaScript Cookies

- **What are Cookies?**
  - Data, stored in small text files, on you computer.
  - "how to remember information about the user"
    - When a user visits a web page, his name can be stored in a cookie.
    - Next time the user visits the page, the cookie "remembers" his name.
  - Saved in name-value pairs like

```
username = John Doe
```

- **Create a Cookie with JavaScript**

```
document.cookie="username=Jone Doe"'
```

```
document.cookie="username=Jone Doe; expires=Thu, 18 Dec 2013 12:00:00 UTC"'
```

```
document.cookie="username=Jone Doe; expires=Thu, 18 Dec 2013 12:00:00 UTC; path=/";'
```

JavaScript Cookies (cont'd)

- **Read a cookie with JavaScript**

```
var  x = document.cookie;
```

  - **document.cookie** will return all cookies in one string much like: cooki1=value; cookie2=value;cookie3=value.

- **JavaScript Cookie Example** *Try it!*
  - A function to set a cookie : stores the name of the visitor in a cookie variable.
  - A function to get a cookie : returns the value of a specified cookie.
  - A function to check a cookie value : checks if a cookie is set.