JavaScript, DOM and CSS

Week 5 Session1

Contents of This session

- JavaScript
  - DOM (document object model)
  - DOM API (application programming interface)
  - How to use DOM

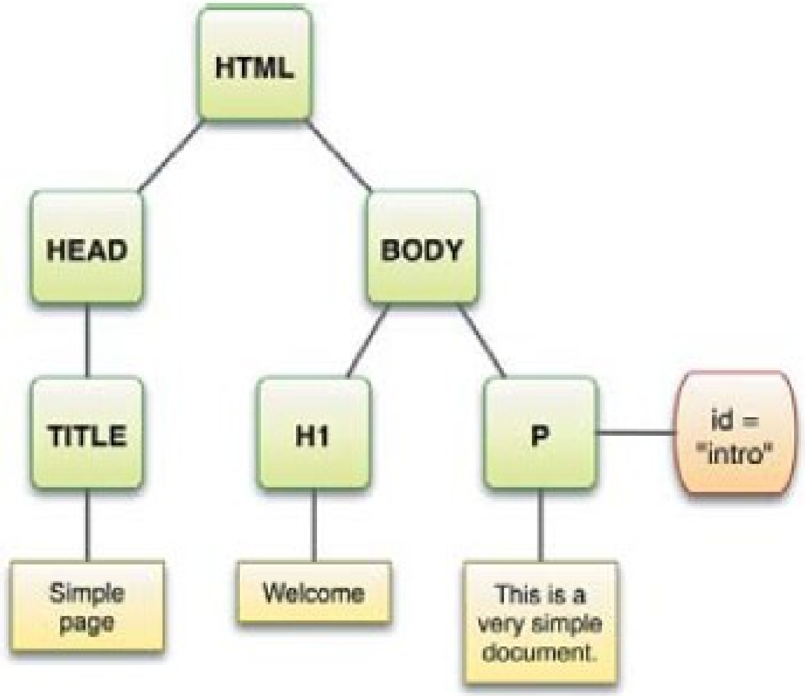
Document Object Model (DOM)

- DOM is a cross-platform and language-independent convention for representing and interacting with objects in HTML, XHTML, and XML documents

- The DOM is an API (application programming interface)
  - It is platform- and language-neutral (accessible in JavaScript, PHP, etc.)
  - As a data structure, it has a tree-based structure
  - It is designed to allow programs and scripts to dynamically *access* and *update* documents

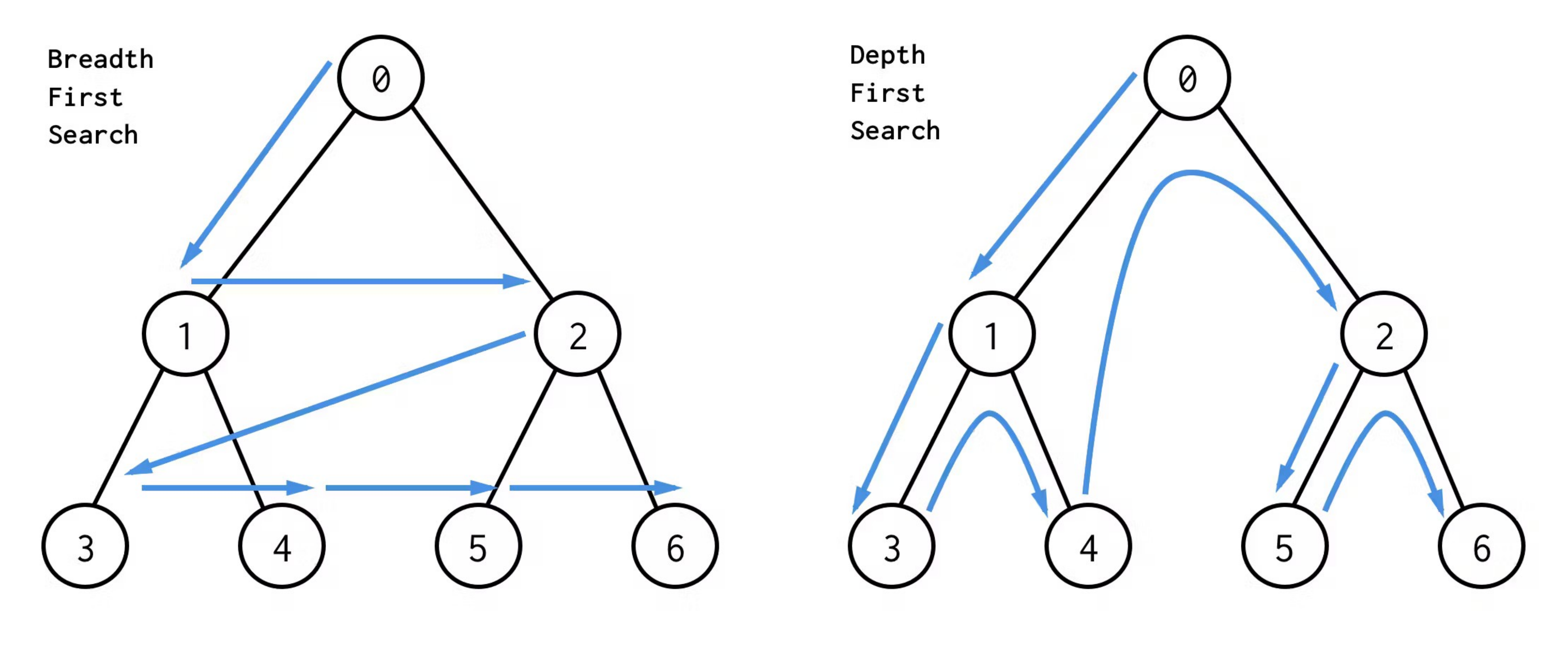
DOM Example – Tree Representation

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
        <head>
        <title>Simple page</title>
        </head>
        <body>
        <h1>Welcome</h1>
        <p id="intro">This is a very simple documen
        </body>
</html>
```



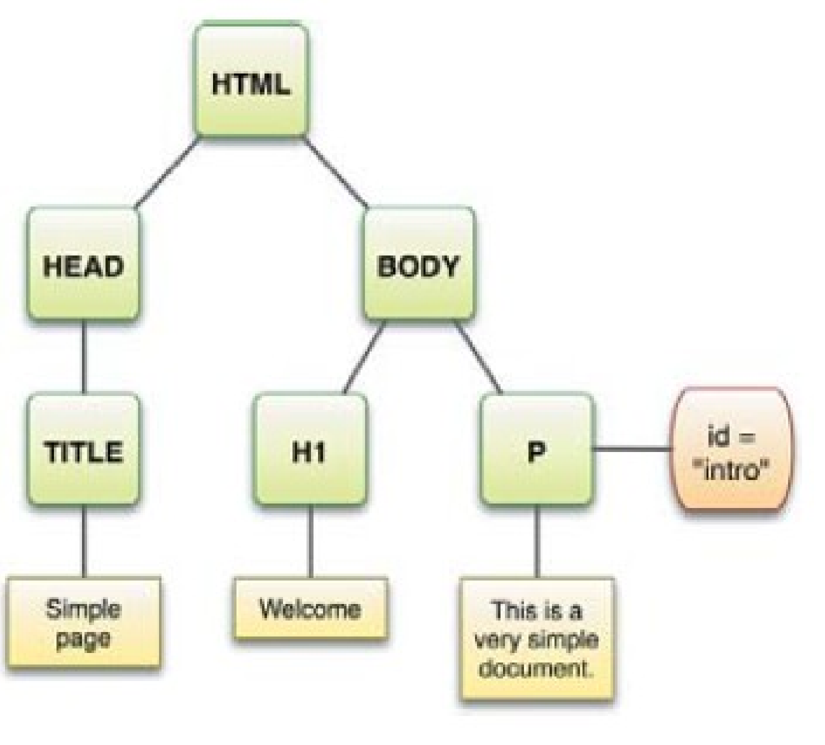
Tree Representation of HTML

DOM Example – Tree Traversal



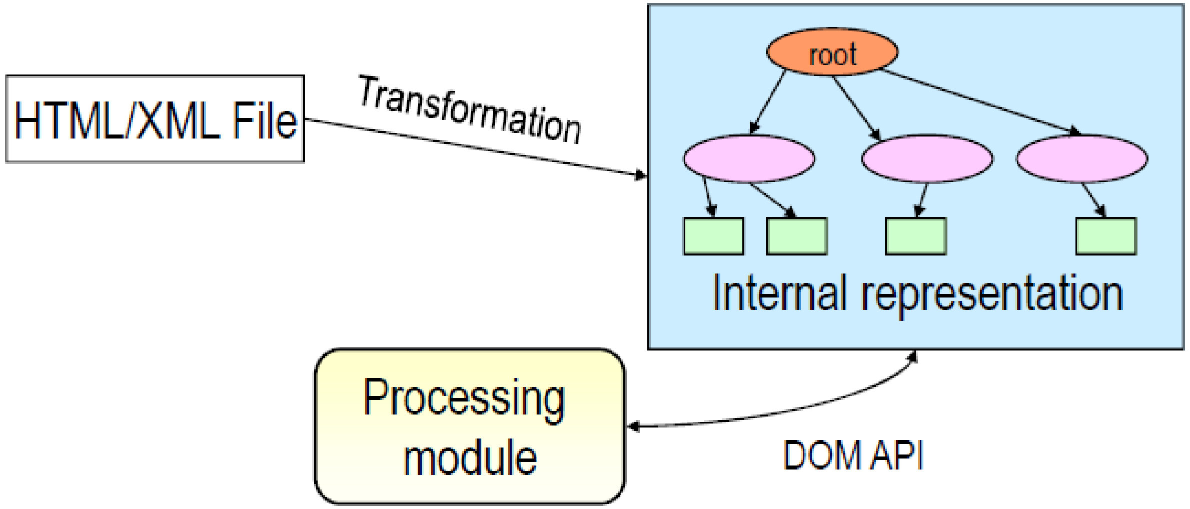
DOM Example

- The html element is the parent of the head element and the body element.
- The head and body elements are siblings.
- The head element is the parent of the title element.
- The title element is a child of the head element.

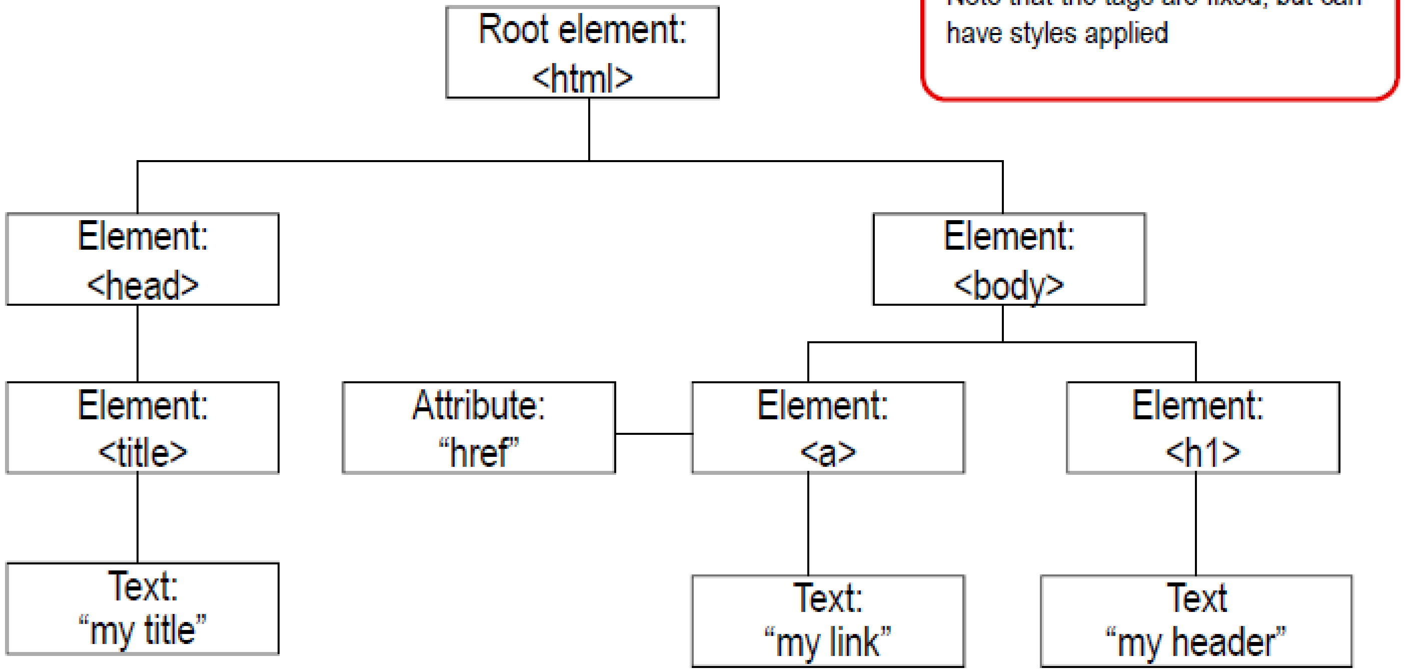


DOM – The Key Concept

- Input: An HTML(XHTML)/XML document
- Internal representation: An in-memory "tree" data
- structure that can be accessed by JavaScript
- Access mode: A set of standard API calls



An HTML Tree

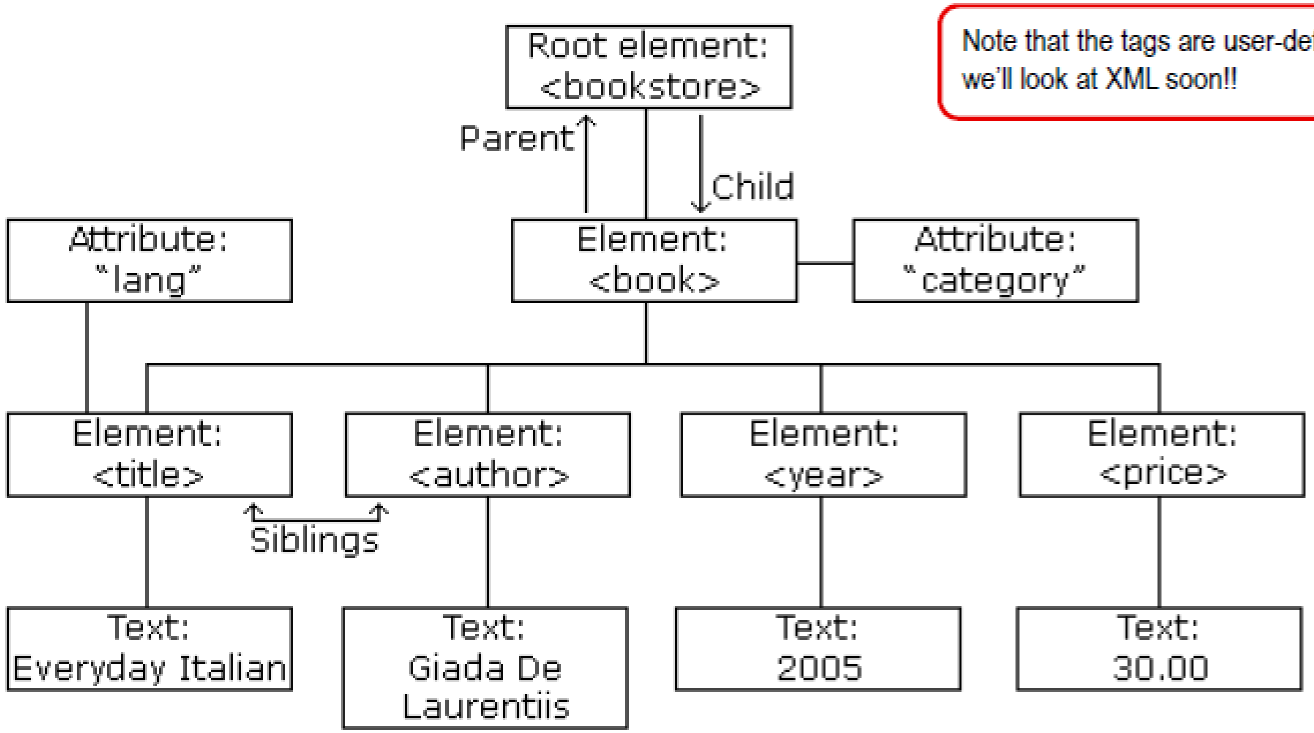


Note that the tags are fixed, but can have styles applied

XML TREE

```
<bookstore>
    <book category=“computer”>
        <title lang=“en”>Everyday Italian</title>
        <author>Giada De Laurentis</author>
        <year>2005</year>
        <price>30.00</price>
    </book>
</bookstore>
```

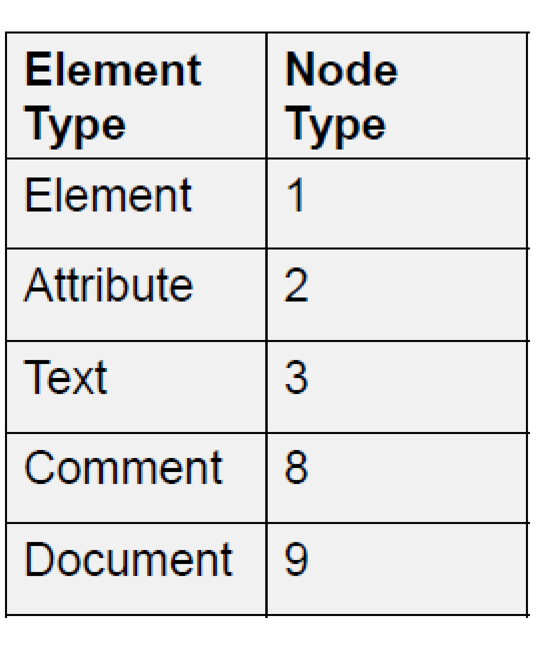
XML TREE



Note that the tags are user-defined – we'll look at XML soon!!

DOM Interfaces

- Everything in a document is a node
  - The entire document is a document node - **document**
  - Every HTML tag is an **element** node
  - The texts contained in the HTML elements are **text** nodes
  - Every HTML attribute is an **attribute** node
  - Comments are **comment** nodes



Element Type	**Node Type**
Element	1
Attribute	2
Text	3
Comment	8
Document	9

A Table of Node Type

DOM Interfaces (Cont’d)

- A **Node Interface** is used to read and write the individual elements in the HTML (or XML) node tree.
- The **childNodes** property of an **element** node (including the **documentElement**) together with a loop construct can be used to enumerate each individual node for processing.
- Main functions of the DOM API
  - To traverse the node tree
  - To access and modify the nodes and their attribute values
  - To insert and delete nodes

Document Object Methods

Method	Description
createAttribute(attributeName)	creates an attribute node with the specified attribute name
createElement(tagName)	creates an element with the specified tagName
createTextNode(text)	creates a text node, containing the specified text
getElementsByTagName(tagName)	returns a (node) list of all nodes with the specified tag name
getElementsByName(name)	returns a list of all nodes with the specified name attribute
getElementById(id)	returns the node with the specified id attribute

Example of Document object methods and properties(more on W3schools)

Node Object Properties and Methods

Property	Description
childNodes	returns a NodeList containing all the child nodes for this node
firstChild	returns the first child node for this node
lastChild	returns the last child node for this node

Method	Description
appendChild(newChild)	appends the node newChild at the end of the child nodes for this node
cloneNode(boolean)	returns an exact clone of this node. If the boolean value is set to true, the cloned node contains all the child nodes as well
hasChildNodes()	returns true if this node has any child nodes

Example of Node object methods and properties(more on W3schools)

Style Object Properties

- To access a style object from a node object

```
NodeObject.style
```

- Then you can read/write a property of your choice

Read: `NodeObject.style.property_Name;`

Write: `NodeObject.style.property_Name = Your_Value;`

Property	Description
fontSize	sets or returns the font size of the text
backgroundImage	sets or returns the background image of an element
borderStyle	sets or returns the style of an element's border.

Example- Use DOM to update HTML

- Toggle Border and Background
- Create and add a new element to a page
- Dynamically change the text style
- Hide the content or display content

Exercise

- JavaScript Dom exercise

End of The Session 1

Week 5