WSA | Forward looking IT finishing school

Document Object Model

JavaScript

Introduction

(DOM)

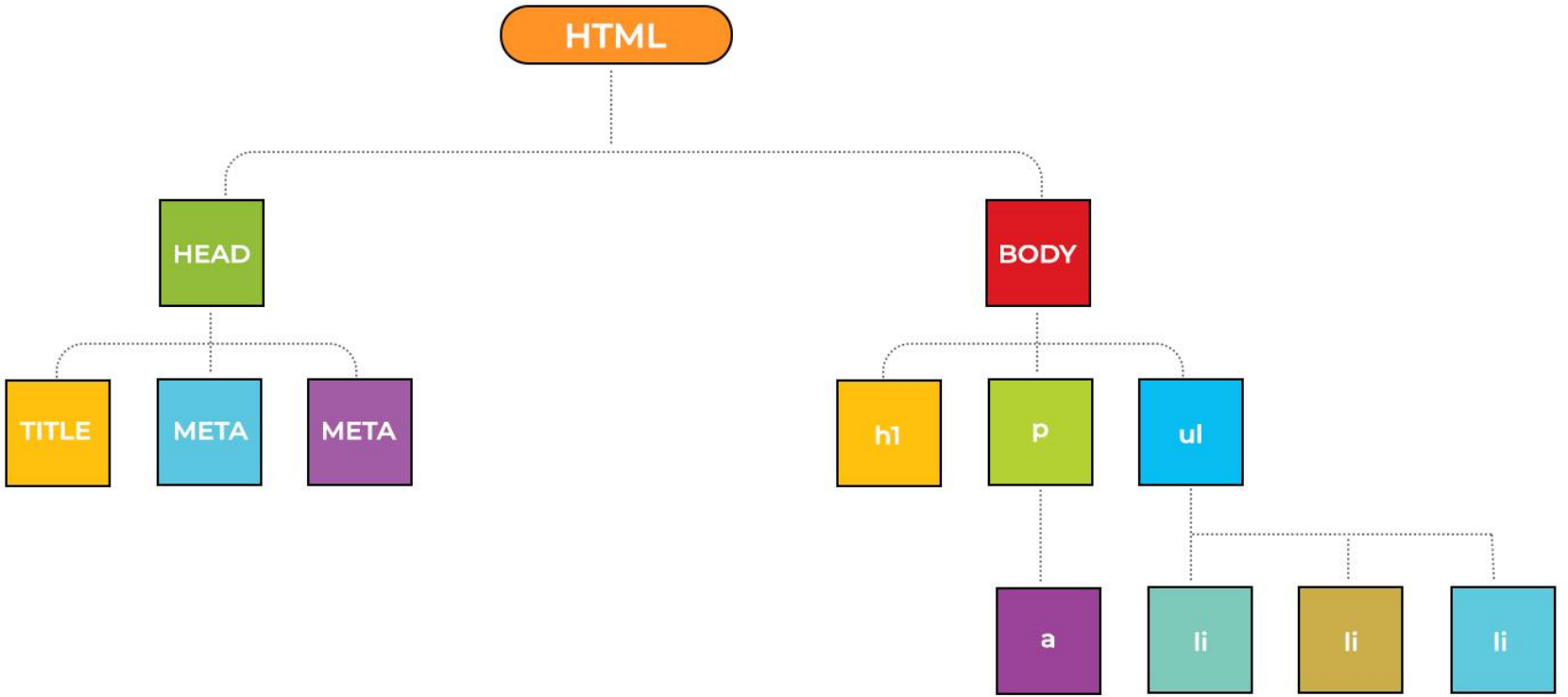
What is DOM?

- The Document Object Model (DOM) is a cross-platform and language-independent application programming interface.
- The DOM, is the API through which JavaScript interacts with content within a website.
- The DOM API is used to access, traverse and manipulate HTML and XML documents.
- The DOM is a W3C (World Wide Web Consortium) standard.

What is DOM Tree?

- The DOM treats an HTML, XHTML, or XML document as a tree
- structure
- Each node is an object representing a part of the document
- User case of HTML, XML and XHTML are different but DOM acts as a common standard

HTML DOM Tree Objects



Types of DOM nodes

- **Element nodes (HTML tag):** Can have children / attributes.

- **Text nodes:** Text nodes are contained within element nodes and cannot have child elements. Text nodes means all the text within nodes.

- **Attribute nodes (Attribute / Value pairs):** Text / Attributes are children of element node, they cannot further have children or attributes

DOM standards

DOM level 0:

- Earliest implementation of DOM
- There existed no standard when its was implemented in some of the major browsers before 1998

DOM level 1

- It was recommended by W3C in 1998
- It will provide complete model for an entire HTML or XML document, to change any portion of the document

DOM level 2

- It was published in 2000 by introducing additional features – getElementById, Event model,

  XML namespace (Avoiding name conflicts), Stylesheet

DOM standards

DOM level 0:

- Earliest implementation of DOM
- There existed no standard when its was implemented in some of the major browsers before 1998

DOM level 1

- It was recommended by W3C in 1998
- It will provide complete model for an entire HTML or XML document, to change any portion of the document

DOM level 2

- It was published in 2000 by introducing additional features – getElementById, Event model,
- XML namespace (Avoiding name conflicts), Stylesheet

DOM standards

DOM level 3

- It was published in 2004
- Added support for Xpath (Navigating between multiple nodes in XML)
- Keyboard event handling

DOM level 4

- It was published in 2015
- Latest version of DOM

Methods

(APIs in DOM)

DOM methods

- HTML DOM **Methods** are actions we can perform on HTML elements
- HTML DOM **Properties** are values that we can set or change

```
<script>

document.getElementById("ex").innerHTML = "Hello World!";

</script>
```

- getElementById -> Method
- innerHTML -> Property

DOM Methods

getElementById()

- Accesses any element on the page via its ID attribute
- A fundamental method within the DOM for accessing elements on the page
- This method will returns single element

innerHTML

- The innerHTML is used to get and replace the content of HTML elements.

DOM Element

getElementsByName(name)

Returns an list with the given name attribute

```
var allParagraphs = document.getElementsByName("number");
```

getElementsByTagName(name)

Returns a list of elements with the given tag name.

```
var allParagraphs = document.getElementsByTagName("p");
```

Element by class name

getElementsByClassName()

Returns a list of all elements with given class name

```
var elements = document.getElementsByClassName("example");
```

DOM Example

```
<script>
function GetById(){
   document.getElementById("div1").innerHTML=document.getElementById("num").value;
}
function GetByName()
{
    var nums= document.getElementsByName("number");
    var res='';
    for(var i=0; i<nums.length; i++)
    {
        res+=nums[i].value +"<br/>";
        document.getElementById("div1").innerHTML= res;
    }
  }
 </script>
```

DOM Example

```
<body>
    <form id="form1">
    <div>
        <input type="text" id="num" value="12345" />
        <input type="text" name="number" value="num1" />
        <input type="text" name="number" value="num2" />
        <input type="text" name="number" value="num3" />
    </div>
    <br />
    <div id="div1" style="border-style: solid; border-width: thick; border-color: blue;
        width: 515px; height: 100px;">
          Div Section
    </div>
    <br />
    <input type="button" onclick="GetById();" value="Get Element By Id" />
    <input type="button" onclick="GetByName();" value="Get Element By Name" />
    </form>
</body>
```

Elements by CSS Selectors

querySelector()

Returns the first match of the passed selector string

```
firstMatchElement = document.querySelector(“.example”);
```

querySelectorAll()

Returns a NodeList of DOM elements that match the query

```
allMatchElements = document.querySelectorAll(“.example");
```

Exercise

- Write a JavaScript program to modify the text-align, font-size, font-family of heading1 using getElementById
- Write a JavaScript program to change the background color of all the tag
- Write a JavaScript to add the text shadow in all paragraphs in the given essay

Some useful tips:

- `<element-name>.style.textShadow = "Apx Bpx"` will set you the shadow

Documents

(Manipulating documents using DOM APIs)

Documents

- Web browser window is represented by window object
- Every window has a document property that refers to document objects
- The document objects represents the web page
- We can access and manipulate HTML using document objects

Changing HTML elements

Methods	Description
`element.innerHTML()`	Change the inner HTML of an HTML element
`element.attribute()`	Change the attribute value of an HTML element
`element.setAttribute()`	Changes the attribute value of an HTML element

Adding and Deleting Element

Methods	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()`	Write into the HTML output stream
`document.insertBefore()`	Insert before an HTML element

Adding Event Handlers

Methods	Description

`Document.getElementById(id).onclick = function()`
`{`
 	
`// Code here…`
 	
`}`
	Adding an event hander on mouse clicking

HTML DOM – Handling images

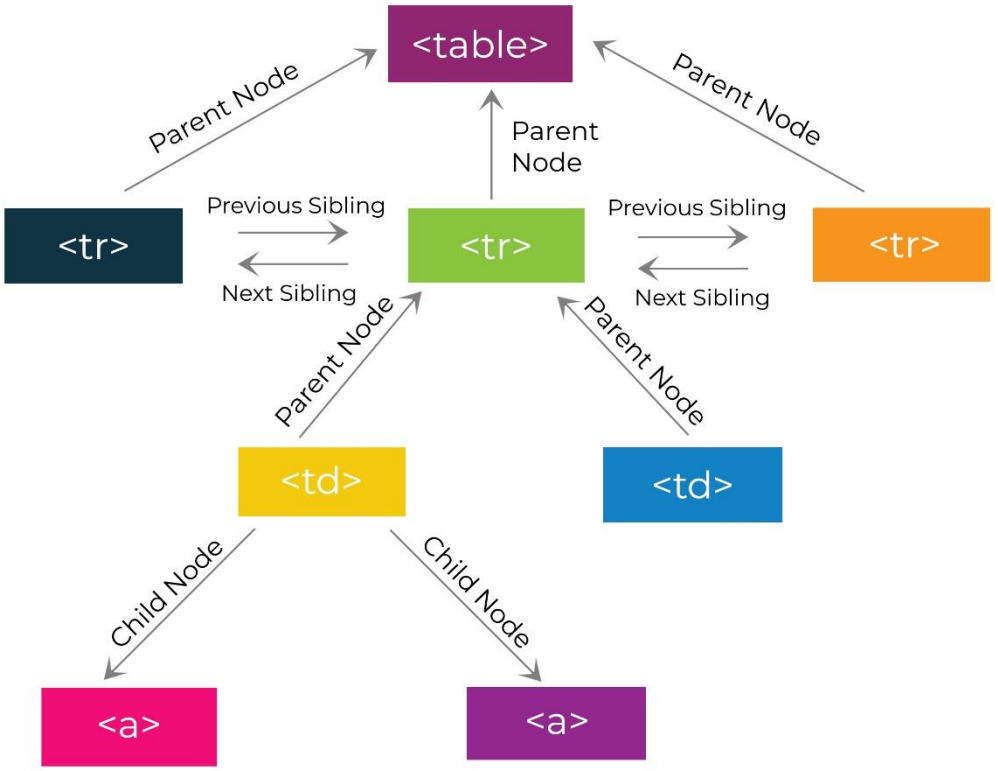
```
<html>
<script>
function imgFunc()
{
    document.getElementById("myimage").src = "jobsold.jpg";
}
</script>

<body>
    <img id="myimage" src="jobsyoung.jpg" width=500 height=200>
    <button onclick="imgFunc()">Click Here to change image</button>
</body>
</html>
```

HTML DOM – Table manipulation

- The Table object of the DOM supports dynamically generate a table or certain rows/columns
- We can access a <table> element by using getElementbyId()

Method	Description
`createCaption()`	Creates an empty <caption> elements and adds it to the table
`deleteCaption()`	Removes the first caption element from the table
`insertRow()`	Inserts new row in the specific index
`deleteRow()`	Removes a row from the table
`insertCell()`	Inserts new column in the specific row
`createTHead()`	Creates an empty <thead> elements and adds it to the table
`deleteTHead()`	Removes the <thead> element from the table
`createTFoot()`	Creates an empty <tfoot> elements and adds it to the table
`deleteTFoot()`	Removes the <tfoot> element from the table

HTML DOM – Table - Relationship view



Exercise

- Modify IPL table manipulation program with following options:
  - Taking user input for new row addition (instead of hard-coded value)
  - Adding new column in all rows called “City” and populate the city name the team belongs to
  - Add option to delete the column by taking the column index from the user
  - Insert table head as “IPL Team details” with appropriate column names (Name, Team, Captain etc..)

Some useful tips:

- `<table-name>.rows.length` will give you number of rows
- `<table-name>.rows[0].cells.length` will give you number of columns

JavaScript Animation

(DOM – Animation APIs)

JavaScript Animation

- The JavaScript animation is implemented as gradual changing of DOM element styles or canvas objects
- The whole process is split into pieces, and each piece is called by timer
- An animation is created by replacing one Image frame with another at speed such that it appears to be a moving Image
- Animations can be created using JavaScript by using a timer which replaces one image frame with another
- The two timer function setTimeout() and setInterval() to execute JavaScript codes at set intervals

DOM Animation

Method	Description
`setTimeout(function,duration)`	This function calls function after duration milliseconds from now
`setInterval(function,duration)`	This function calls function after every duration milliseconds
`clearTimeout(timeout)`	This function calls clears any timer set by the setTimeout() functions

Some useful tips:

- `<element-name>.style.left = A px` sets the left position of the element
- `<element-name>.style.right = A px` sets the right position of the element
- `<element-name>.style.top = A px` sets the top position of the element
- `<element-name>.style.bottom = A px` sets the top position of the element

Exercise

- Write a JavaScript program to move two small squares inside one big square in a random manner. User should be able to start and stop this animation using button based events

Some useful tips:

```
Math.floor(Math.random() * Math.floor(max)) will give you a
random number that is less than max value
```

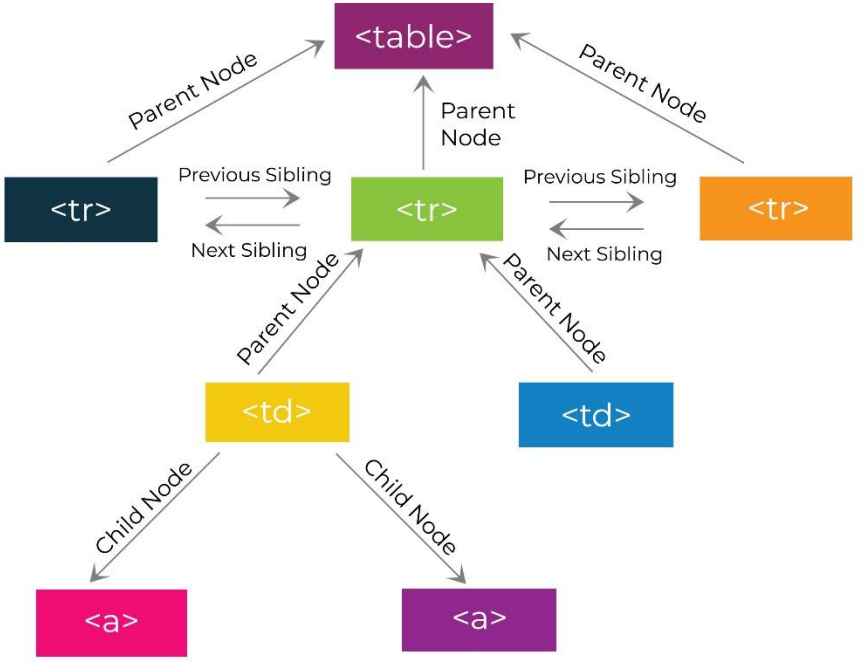
Node & NodeList Handling

DOM – Node Handling APIs

DOM Nodes

- In the context of DOM, every entity in an HTML or XML document, is referred as a Node
- In JavaScript, all the Nodes are represented as Objects. They have their own properties and methods
  - <Doctype html> is DocumentType Node
  - <head> is an Element Node, so are meta, title, body, h1, p, ul and li
- Text contents like the title text 'This is a simple HTML document' is a text Node
- Some of the Nodes may have children. Like <ul> has <li> as children
- Body has <h1>, <p>, as children

DOM Node – Tree representation - Revisit



Navigation between nodes

There are some node properties to navigate between nodes:

- `parentNode`
- `childNodes[node number]`
- `firstChild`
- `lastChild`
- `nextSibling`
- `previousSibling`

Node creation - Example

```
// Creates a new em element
var newnode = document.createElement("em");

// To add text to the em element
var node = document.createTextNode("Text to add");

// Appending the text node to em element
newnode.appendChild(node);

// Appending new element to existing element
var elem = document.getElementById("p");
elem.appendChild(newnode);
```

Node Creation - Example

```
<body>
<p>Click the button to make a new Button element</p>
<button onclick="myFun()">Click</button>
<script>
function myFun() {
    var btn = document.createElement("BUTTON");
    var t = document.createTextNode("NewButton");
    btn.appendChild(t);
    document.body.appendChild(btn);
}
</script>
</body>
```

DOM Node - Methods

Method	Description
`appendChild()`	The appendChild() method will add new element as last child of parent , if we want to insert before last child of parent then insertBefore() can be used. Syntax : parentElem.insertBefore(elem, nextSibling);
`removeChild()`	Remove child node from parent node. Syntax : parentElem.removeChild(elem);
`replaceChild()`	Replace the child element of parent Element, referenced by current Element with the element. Syntax : parentElem.replaceChild(elem, currentElem);

DOM Node List

- The Node List object represents an ordered collection of nodes, indexed by number (starting from zero).
- A Node List is not an array. For example the getElementsByTagName() method returns a node list. The nodes can be accessed by an index number.
- The length property defines number of nodes in the list

```
var allParagraphs  = document.getElementsByTagName("p");
var firstParagraph = allParagraphs[0];
```

Exercise

- Write a JavaScript function to manipulate an unordered list with multiple following options:
  - Create multiple lists
  - Dynamically assign ID for each of the lists created
  - In each of the lists created support the following operations:
    - ✓ Insert (Before)
    - ✓ Append
    - ✓ Replace
    - ✓ Remove

WSA | Forward looking IT finishing school

Thank you

WebStack Academy

#83, Farah Towers,
1st Floor, MG Road,
Bangalore – 560001

M: +91-809 555 7332
E: training@webstackacademy.com

WSA in Social Media: