OBJECT™
TECHNOLOGIES

HTML DOM

What will be covered

OBJECT TECHNOLOGIES

- What is DOM
- HTML DOM API
- HTML DOM node tree
- Classification of Nodes
- Node Relationships
- HTML DOM Document Object
- Accessing HTML elements
- Whitespace nodes

What is DOM

OBJECT

- The Document Object Model (DOM) is the data representation of the objects that comprise the structure and content of a document on the web.
- The Document Object Model (DOM) is a programming interface for web documents. It represents the page so that programs can change the document structure, style, and content. The DOM represents the document as nodes and objects; that way, programming languages can interact with the page
- A web page is a document that can be displayed in the browser window. It can be modified with a scripting language such as JavaScript.
- The DOM is a W3C (World Wide Web Consortium) standard.
- The DOM defines a standard for accessing documents:

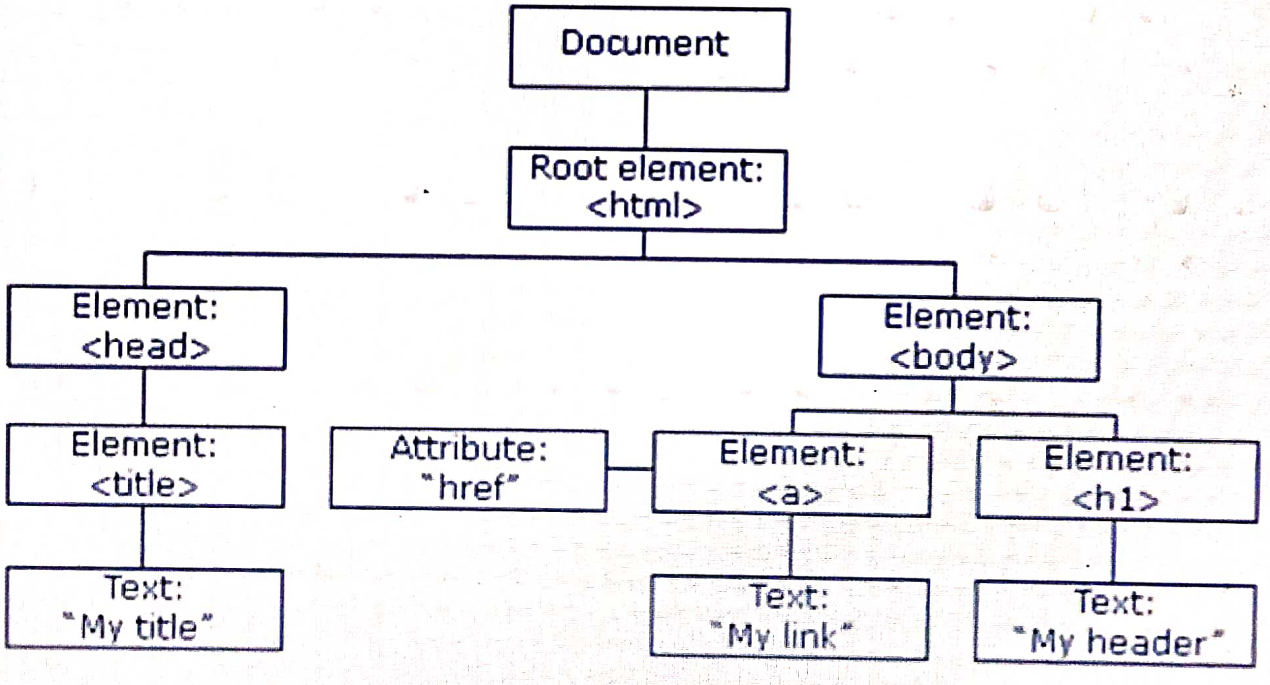
HTML DOM API

- HTML DOM is 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
- A **property** is a value that you can get or set (like changing the content of an HTML element).
- A **method** is an action you can do (like add or deleting an HTML element).

HTML DOM node tree

OBJECT

- When a web page is loaded, the browser creates a **D**ocument **O**bject **M**odel of the page.
- The **HTML DOM** model is constructed as a tree of **Objects.**
- Any node can have any no of child with only one root.
- Nodes having a common parent are called siblings



HTML DOM node tree

OBJECT

- In DOM terminology, all elements of an HTML document are nodes, including the entire document and all the HTML tags contained within it. Text constitutes text nodes, comments make up the comment nodes, while HTML attributes make up the attribute nodes.
- <html> is a root node within which all other nodes are contained.
- <head> and <body> are the children nodes of the <html> root node.
- the <head> node is the parent of the <title> node. The <a> and <h1> nodes are the children of the <body> node.
- The text inside elements forms text nodes. A text node contains only a string. It may not have children and is always a leaf of the tree.

Classification of Nodes

OBJECT

- All nodes contained in a document are associated with a node type. This can be examined using nodeType property.

Node Type	Value	Example
ELEMENT_NODE	1	The <body> element
TEXT_NODE	3	Text that is not part of an element
COMMENT_NODE	8	<!-- an HTML comment -->
DOCUMENT_NODE	9	A Document node.
DOCUMENT_TYPE_NODE	10	A DocumentType node, such as <!DOCTYPE html>.
ATTRIBUTE_NODE	2	An Attribute of an Element.

- The most important are element node, text node and attribute node for DOM manipulation

Node Relationships

OBJECT

- Node relationships are in the form of parents, child and sibling and this can be observed in the document tree.
- Relationships tend to be hierarchical, with parent nodes above the children nodes, and sibling nodes on the same level.
  - Root node: This is the topmost node in a node.
  - Parent node: This is any node in the tree that has a node below it, all nodes except the root have a parent node.
  - Child node: Any node with one node above it.
  - Sibling nodes: All nodes sharing a parent node.
  - Leaf node: This is a node that has no children.
  - Grandchildren: These are the child nodes of a child node.

HTML DOM Document Object

OBJECT

- The document object represents your web page.
- If you want to access any element in an HTML page, you always start with accessing the document object.
- Documents object can be used for Finding HTML Elements, finding HTML objects as collections
- Documents object can be used even for Adding and Deleting Elements.

Accessing HTML Elements

OBJECT

- Finding HTML Element by Id :

  Most easiest way to find HTML element using id attribute value, if element is found it returns the

```
var myElement = document.getElementById("intro");
```

- Finding HTML Elements by Tag Name

  Following code finds all the p elements in the document

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

- Finding HTML Elements by Class Name

  If you want to find all HTML elements with the same class name, use getElementsByClassName().

```
var x = document.getElementsByClassName("intro");
```

Accessing HTML Elements

OBJECT

- Finding HTML Elements by CSS Selectors

  HTML elements that matches a specified CSS selector (id, class names, types, attributes, values of attributes, etc), use the querySelectorAll() method.

```
var x = document.querySelectorAll("p.intro");
```

- Finding HTML Elements by HTML Object Collections

  Few HTML elements are accessible as collections.

```
var allLinks = document.anchors;
var text = "";
for (i = 0; i < allLinks.length; i++)
{
    text += allLinks[i].innerHTML;
}
document.getElementById("anchors").innerHTML = text;
```

Whitespace nodes

OBJECT

- The presence of whitespace in the DOM can cause layout problems and make manipulation of the content tree difficult in unexpected ways, depending on where it is located.
- Whitespace is any string of text composed only of spaces, tabs or line breaks (to be precise, CRLF sequences, carriage returns or line feeds). These characters allow you to format your code in a way that will make it easily readable by yourself and other people.
- When trying to do DOM manipulation in JavaScript, you can also encounter problems because of whitespace nodes. For example, if you have a reference to a parent node and want to affect its first element child using Node.firstChild, if there is a rogue whitespace node just after the opening parent tag you will not get the result you are expecting. The text node would be selected instead of the element you want to affect.
- As another example, if you have a certain subset of elements that you want to do something to based on whether they are empty (have no child nodes) or not, you could check whether each element is empty using something like Node.hasChildNodes(), but again, if any target elements contain text nodes, you could end up with false results.

HTML DOM

As now we have sufficient exposure for javascript language, let us learn about how javascript will be really used in manipulating existing HTML documents. We will understand the tree structure of HTML documents and different types of nodes that are available in the tree structure.

Introduction to HTML DOM

DOM stands for Document Object Model.

The Document Object Model (DOM) is a language and platform independent programming interface which can be used to modify any structured document like HTML or XML.

Using DOM properties and methods content , structure or style of the document can be easily modified by accessing it.

It is standardized by the World Wide Web Consortium (W3C).

This model is supported by almost all the modern browsers.

HTML DOM

HTML DOM is a way to represent the webpage in the structured hierarchical way so that it will become easier for programmers and users to glide through the document.

With DOM, we can easily access and manipulate tags, IDs, classes, Attributes or Elements using commands or methods provided by Document object.

In simple meaning, HTML DOM allows to access and modify structure, style and content of HTML page dynamically.

HTML Node and Tree Structure

The DOM represents an HTML or XML document as a hierarchy of nodes. Consider the following HTML document:

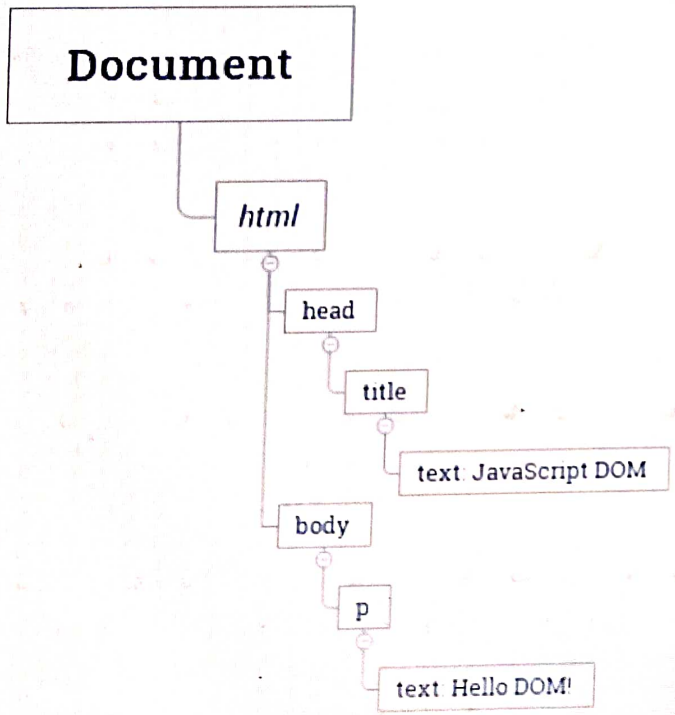
```
<html>
<head>
   <title>[redacted]</title>
</head>
<body>
    <p>[redacted]</p>
</body>
</html>
```

HTML Node and Tree Structure

The DOM represents an HTML or XML document as a hierarchy of nodes. Consider the following HTML document:

```
<html>
<head>
   <title>[redacted]</title>
</head>
<body>
    <p>[redacted]</p>
</body>
</html>
```

The following tree represents the above HTML document:



In this DOM tree, the document is the root node. The root node has one child which is the <html> element.

The <html> element is called the document element.

Each document can have only one document element. In an HTML document, the document element is the <html> element.

Each markup can be represented by a node in the tree.

In this DOM tree, the document is the root node. The root node has one child which is the <html> element.

The <html> element is called the document element.

Each document can have only one document element. In an HTML document, the document element is the <html> element.

Each markup can be represented by a node in the tree.

The text in the element node creates a separate node which is child of its parent element.

Node Types

Each node in the DOM tree is identified by a node type. JavaScript uses integer numbers to determine the node types.

Following table shows most frequently needed types of nodes and their properties.

Sr No	Node Type	nodeName property	nodeValue property	nodeType property
1	Document	#document	null	9
2	Element	tag name in capitals	null	1
3	Attr	attribute name	attribute value	2
4	Comment	#comment	comment text	8
5	Text	#text	content of node	3

A node has few important properties: nodeName , nodeValue and nodeTye that provide specific information about the node. The values of these properites depends on the node type. For example, if the node type is the element node, the nodeName is always the same as element's tag name and nodeValue is always null.

Node and Element

A node is a generic name of any object in the DOM tree. It can be any built-in DOM element such as the document. Or it can be any HTML tag specified in the HTML document like or <p>

Node Relationships

Any node has relationships to other nodes in the DOM tree. The relationships are the same as the one described in a traditional family tree.

- Every tree will have only one root node which does not have parent node. In case of HTML document it is <HTML> which is acting as root node.
- Any node will have only one parent node
- Any node will have any(n) no of children nodes
- Nodes having same parent are called sibling nodes.
- Some nodes may not have any child nodes which are called as leaf nodes

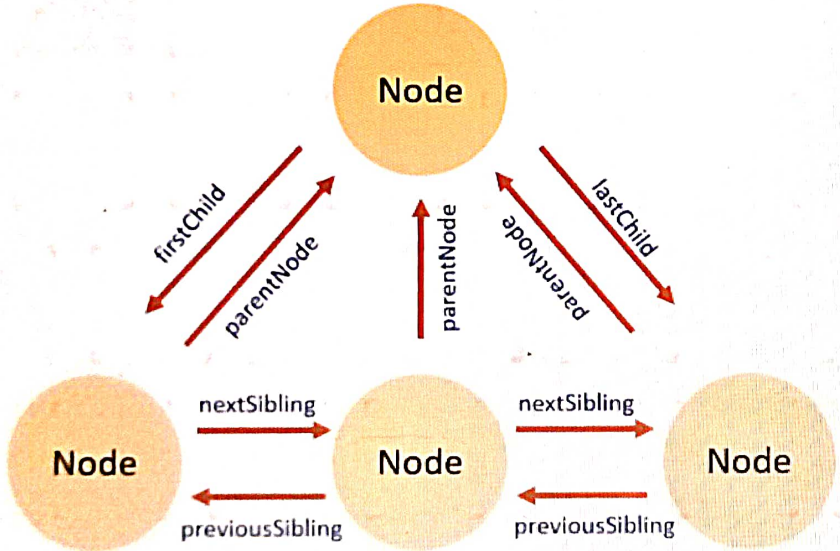
Node and Element

A node is a generic name of any object in the DOM tree. It can be any built-in DOM element such as the document. Or it can be any HTML tag specified in the HTML document like or <p>

Node Relationships

Any node has relationships to other nodes in the DOM tree. The relationships are the same as the one described in a traditional family tree.

- Every tree will have only one root node which does not have parent node. In case of HTML document it is <HTML> which is acting as root node.
- Any node will have only one parent node
- Any node will have any(n) no of children nodes
- Nodes having same parent are called sibling nodes.
- Some nodes may not have any child nodes which are called as leaf nodes.



Accessing HTML Elements

Structurally, the Document Object Model consists of nodes, with each node representing content in the web document. It gives developers a way of representing everything on a web page so that the contents of the web page is accessible via a common set of properties and methods.

Following methods are used to access the node in a way HTML element in node tree ie. HTML document

1. Accessing elements by ID

This method is used when we need to access the element directly from the document

Accessing HTML Elements

Structurally, the Document Object Model consists of nodes, with each node representing content in the web document. It gives developers a way of representing everything on a web page so that the contents of the web page is accessible via a common set of properties and methods.

Following methods are used to access the node in a way HTML element in node tree ie. HTML document

1. Accessing elements by ID

This method is used when we need to access the element directly from the document.

Javascript provides a document.getElementById() method, which is the easiest way to access an element from the DOM tree structure. It will return the element that has the ID attribute with the specified value.

document.getElementById("ID");

Example :

```
alert(document.egtElementById("mypara").innerHTML)
```

Above statement displays the text written in the element having id as "mypara".

2. Accessing elements by tag name

The getElementsByTagName() is one of the method exposes for accessing nodes directly. This method takes a tag name as argument and returns a collection of all the nodes it finds in the document that are a sort of tag.

E.g.

```
var cnt = document.getElementsByTagName("p");
alert(cnt.length);
```

Note : cnt is collection of nodes that should be iterated. Even if document consists of only one paragraph, it still returns collection with only one element.

3. Accessing elements by class value

The getElementsByClass() method works same like getElementById() method, and it will returns a collection of all elements in the document with the specified class name.

E.g.

```
var tmpClass = document.getElementsByClassName("testClass");
```

3. Accessing elements by class value

The getElementsByClass() method works same like getElementByld() method, and it will returns a collection of all elements in the document with the specified class name.

E.g.

```
var tmpClass = document.getElementsByClassName("testClass");
alert(tmpClass.length);
tmpClass[1].innerHTML ="Second Paragraph";
```

4. Selecting elements by query

querySelector() method let you enter a CSS selector as an argument and return the selected elements as DOM elements.

E.g.

```
document.querySelector("#testQuery").innerHTML ="First Paragraph";
```

Note : The argument for this method is any valid CSS selector.

5. Selecting all elements by query

The querySelectorAll() method returns a collection filled with the matching elements in source order.

E.g

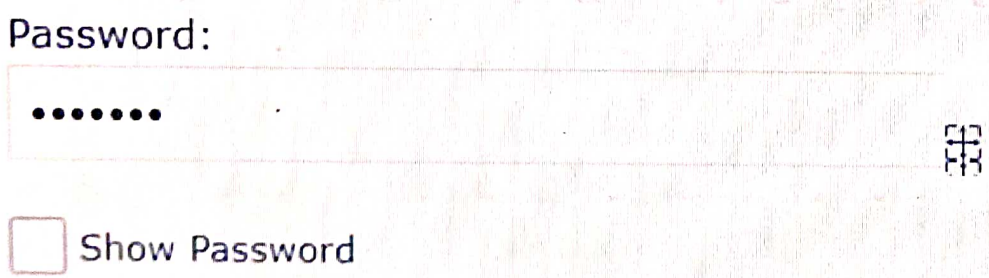
```
var elements = document.querySelectorAll("div,p");
elements[1].innerHTML = "content changed";
```

The second element in the collection is modified with the content

Assignments

1. Create a Web page which contains an image. When the user clicks on the image, the image should start scrolling. Again when he brings mouse over the scrolling image then display the original static image.
2. Write a program to access the HTML tags using DOM concept in the following manner:
    1. Access the contents of all paragraph tags and display them in tabular format on the click event of a button
    2. Change the color of all heading tags on the click event of a button.
    3. Change the background color of all text boxes on the click event of button.
3. Show shadow of the text "Hello World" on mouse over and hide the shadow of the test on mouse out.

Assignments

1. Create a Web page which contains an image. When the user clicks on the image, the image should start scrolling. Again when he brings mouse over the scrolling image then display the original static image.
2. Write a program to access the HTML tags using DOM concept in the following manner:
   1. Access the contents of all paragraph tags and display them in tabular format on the click event of a button
   2. Change the color of all heading tags on the click event of a button.
   3. Change the background color of all text boxes on the click event of button.
3. Show shadow of the text "Hello World" on mouse over and hide the shadow of the test on mouse out.
4. Display 3 buttons on the page having a label of color. Clicking of the button should change the background color of the document.
5. 

When user checks the checkbox, password gets displayed. When checkbox is unchcked, password gets hidden.