UNIT : 1

Document Object Model

Reference :- HTML, JAVASCRIPT, DHTMAL AND PHP by IVAN BAYROSS
Chapter :- 9 : The Javascript Document Object Model

DOM : Introduction

- HTML page is rendered (painted) in a browser.
- Browser assembles all elements(objects) contained in HTML page, downloaded from web server.
- Once rendered browser no longer recognize individual HTML elements (objects).
- To create interactive web page browser should continue to recognize individual HTML elements
- Browser can access properties of these objects.

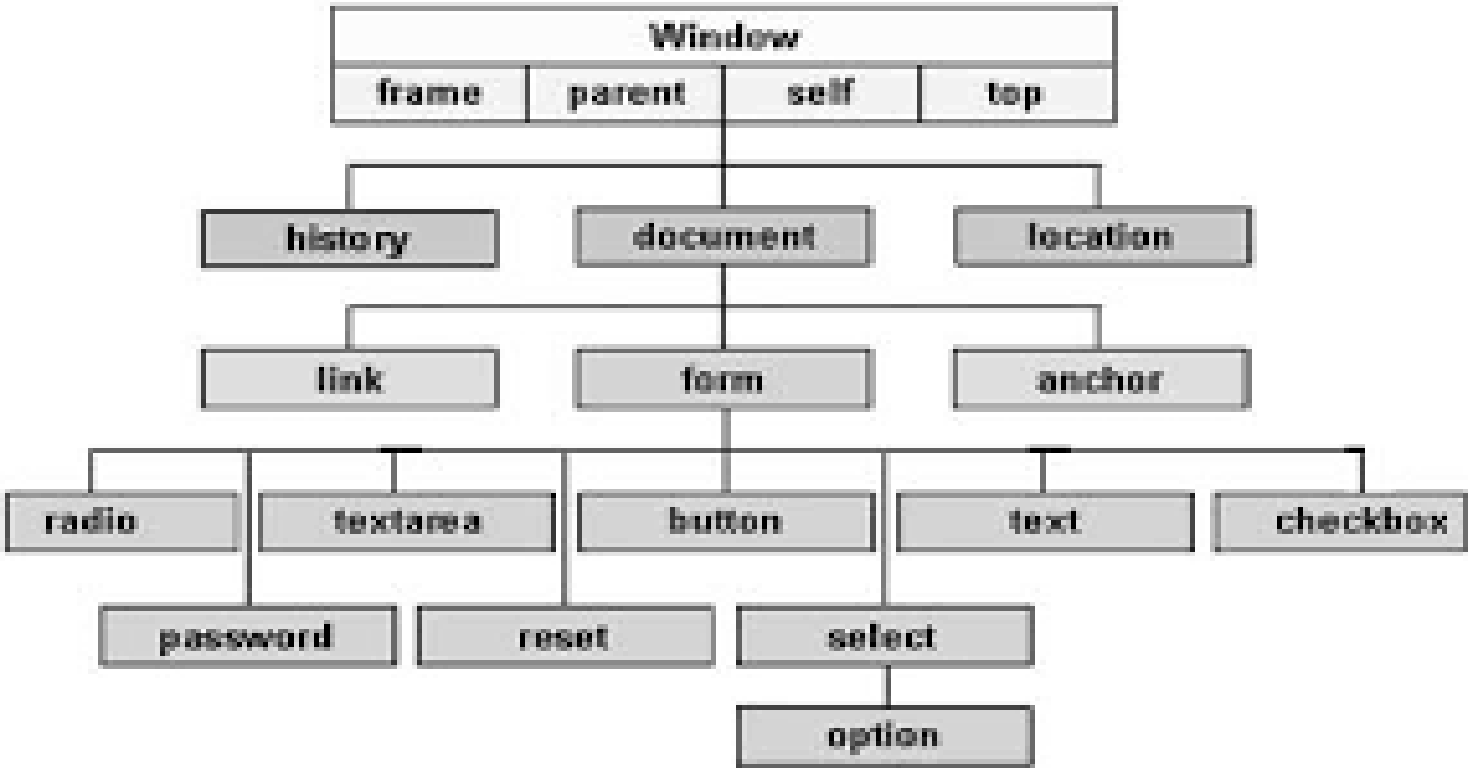
DOM : Introduction

- Javascript enabled browser are capable of recognizing individual elements of HTML
- Javascript enabled browser uses DOM (Document Object Model).
- HTML objects (collection of web page elements) belongs to DOM and have descending relationship with each other.

DOM : Introduction

- Top most object in DOM is Navigator (i.e. Browser)
- Next level in DOM is browser's Window.
- Next level in DOM is the Document displayed in browser's window.
- If document contains Form then next level is Form.
- Next level contains individual elements on a Form.
  - E.g. text boxes, radio buttons check boxes etc.
- Javascript's DOM is referred to as an ***instance hierarchy***

DOM : Introduction



DOM : Introduction

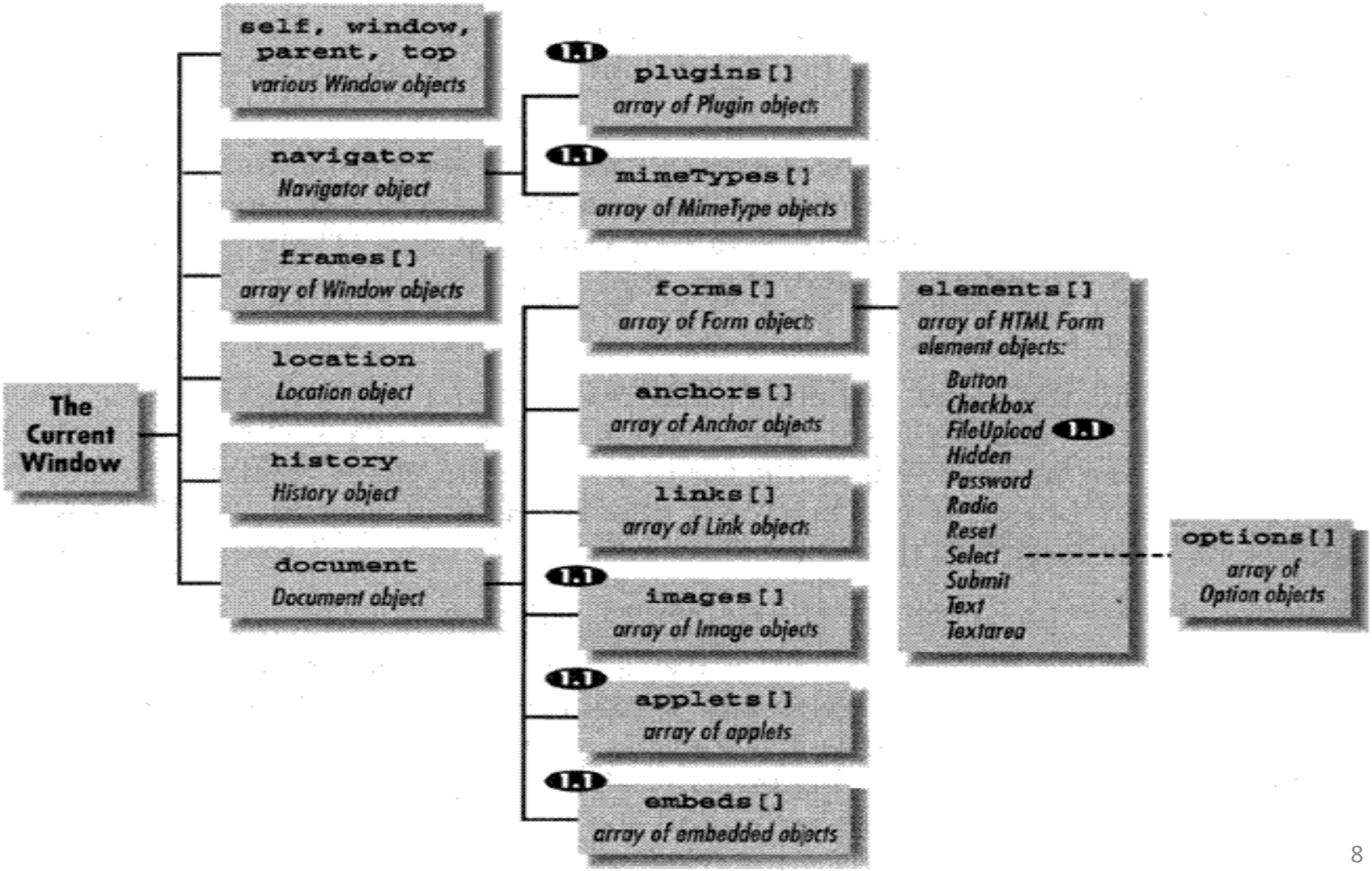
- Window
  - Document
    - Link
    - Anchor
    - Form
      - textbox
      - textarea
      - radiobutton
      - checkbox
      - select
      - button

DOM : Introduction

- Instance
  - JavaScript enabled browser assembles HTML elements in memory prior being rendered in browser window.
  - If document doesn't have Anchors described, Anchors object will exist but it will be empty.

- Hierarchy
  - In addition to DOM, other objects recognized by JavaScript enabled browser are plug-ins, Applets and images.
  - Every element of web page rendered in browser are not part of DOM.

DOM : Introduction



JSSS DOM

- Javascript Assisted Style Sheets DOM
  - Uses JavaScript syntax to control document's presentation style.
  - When embedded within <HEAD>…</HEAD> JavaScript DOM takes new set of objects which is added to the standard DOM objects.

JSSS's DOM

- Window
  - Document
    - Tags
      - P
      - DIV
      - SPAN
      - H1 through H6
    - Classes
      - Tag names
    - IDs

JSSS's DOM

(1) getElementsByTagName ()

- Example

(2) getElementById()

- Example

(3) getElementsByClassName()

- Example

(4) getElementsByName() :

- The getElementsByName() method returns a collection of all elements in the document with the specified name (the**value** of the name attribute), as a NodeList object.
- Example

JSSS DOM

- JavaScript understands Objects.
- Objects have
  - Properties → Determines functionality of object.
  - Methods → Allows access to these properties
  - Events → allows JavaScript code to be connected to the object by being mapped to appropriate JavaScript event handlers.

- JavaScript is an Object-based programming language it is not fully object oriented.
- It doesn't support classification, Inheritance, encapsulation and information hiding.

Objects in HTML

- Properties
  - Determines behavior/state of the object.
  - Can be referenced as:
    - ObjectName.PropertyName
    - E.g. textbox → name, size etc.

- Methods
  - To control state of the object at run time.
  - Used to set / get a value of object's property.
  - Can be referenced as:
    - ObjectName.MethodName

Browser Objects

- When web page loads browser automatically creates no. of objects that map to the DOM.

- Javascript object created by [Netscape Communicator] are :

Browser Objects

Object Name	Use
navigator	To access information about the browser that is executing the current script.
window	To access a browser window or a frame within window.
document	To access the document currently loaded into a window.
location	To represent URL. Used to create a URL object, access parts of a URL, or modify an existing URL.
history	To maintain a history of URL's accessed within a window.
event	To access information about the occurrence of an event.
EVENT	Provides constants that are used to identify events.
screen	To access information about size and color depth of a client computer's screen in which current browser is running.

Navigator Object Properties

Property	Description
appCodeName	Returns the code name of the browser
appName	Returns the name of the browser
appVersion	Returns the version information of the browser
cookieEnabled	Determines whether cookies are enabled in the browser
geolocation	Returns a Geolocation object that can be used to locate the user's position
language	Returns the language of the browser
onLine	Determines whether the browser is online
platform	Returns for which platform the browser is compiled
product	Returns the engine name of the browser
userAgent	Returns the user-agent header sent by the browser to the server

Window Object

Method	Description
open(URL)	Opens a new browser window
setInterval()	Calls a function or evaluates an expression at specified intervals (in milliseconds)
setTimeout()	Calls a function or evaluates an expression after a specified number of milliseconds
close()	Closes the current window
clearInterval()	Clears a timer set with setInterval()
clearTimeout()	Clears a timer set with setTimeout()
alert()	Displays an alert box with a message and an OK button
stop()	Stops the window from loading

History Object

Property	Description
Length	Returns the number of URLs in the history list

Method	Description
back()	Loads the previous URL in the history list
forward()	Loads the next URL in the history list
go()	Loads a specific URL from the history list

Example

Other Build-In Object In JavaScript

- 1) The String Object
- 2) The Math Object
- 3) The Date Object

String Object

- **String Properties**

length	Returns the length of a string

String Function :

Method	Description
big()	Displays a string using a big font
blink()	Displays a blinking string (Only supported in Opera 12 and earlier versions)
bold()	Displays a string in bold
charAt()	Returns the character at the specified index (position)
italics()	Displays a string in italic
toLowerCase()	Converts a string to lowercase letters
toUpperCase()	Converts a string to uppercase letters
substring()	Extracts the characters from a string, between two specified indices

String Object

Syntax :
str.substring(*indexA* [, *indexB*])

substring() extracts characters from indexA up to but not including indexB.
In particular:
•If indexA equals indexB, substring() returns an empty string.
•If indexB is omitted, substring() extracts characters to the end of the string.
•If either argument is less than 0 or is NaN, it is treated as if it were 0.
•If either argument is greater than stringName.length, it is treated as if it were stringName.length.
If indexA is larger than indexB, then the effect of substring() is as if the two arguments were swapped; for example, str.substring(1, 0) == str.substring(0, 1).

Date Object in JavaScript

Methods	Description
getDate()	Returns the day of the month as an integer from 1 to 31.
setDate()	Sets the day of the month based on an integer argument from 1 to 31.
getHours()	Returns the hours as an integer from 1 to 23.
setHours()	Sets the hours based on an integer argument from 1 to 23.
getTime()	Returns the number of milliseconds since 1 January 1970 at 00:00:00.
setTime()	Sets the time based on an argument representing the number of milliseconds since 1 January 1970 at 00:00:00.

Math Object in JavaScript

- Properties :

Properties	Description
E	Euler's constant (2.71)
LN10	The Natural logarithms of 10 (roughly 2.302).
LN2	The Natural logarithms of 2 (rougthly 0.693)
PI	The ratio of the circumference of a circle to the diameter of the same circle (roughtly 3.1415)

Math Object in JavaScript

Method	Description
Abs()	Calculates the absolute value of a number.
Ceil()	Returns the next integer greater than or equal to a number.
Cos()	Calculates the cosine of a number.
Floor()	Returns the next integer less than or equal to a number.
Pow()	Calculates the value of one number to the power of a second number –takes two arguments.
Random()	Returns a random number between zero and one.
Sin()	Calculates the sine of a number.
Sqrt()	Calculates the square root of a number.
Tan()	Calculates the tangent of a number.

Include External Javascript file

myScript.js

```
function myFunction()
{
   document.getElementById("demo").innerHTML = "Paragraph changed.";
}
```

Mypage.html

```
<html>
<body>
<h1>External JavaScript</h1>
<p id="demo">A Paragraph.</p>
<button type="button" onclick="myFunction()">Try it</button>
<script src="myScript.js"></script>
</body>
</html>
```

Example

Types of CSS

There are three ways to use CSS:

- External style sheet
- Internal style sheet
- Inline style

External Style Sheet

- An external style sheet is ideal when the style is applied to many pages.
- With an external style sheet, you can change the look of an entire Web site by changing just one file.
- Each page must include a link to the style sheet with the <link> tag. The <link> tag goes inside the head section:

- <head>
  <link rel="stylesheet" type="text/css" href="mystyle.css">
  </head>
- Mystyle.css
- body {
  background-color: lightblue;
  }
  h1 {
  color: navy;
  margin-left: 20px;
  }

Example

Internal Style Sheet

An internal style sheet should be used when a single document has a unique style.

You define internal styles in the head section of an HTML page, inside the <style> tag, like this:

```
<head>
<style>
body {
    background-color: linen;
}
h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
</head>
```

Example

Inline Styles

- An inline style loses many of the advantages of a style sheet (by mixing content with presentation).
- To use inline styles, add the style attribute to the relevant tag. The style attribute can contain any CSS property. The example shows how to change the color and the left margin of a h1 element:
- Example

```
<h1 style="color:blue;margin-left:30px;">This is a heading.</h1>
```

THANK YOU