# JavaScript

# OBJECTIVES

By the end of the session, students should be able to

- ❑ Define DOM
- ❑ Differentiate types of programming
- ❑ Define JavaScript
- ❑ Differentiate between Java and JavaScript
- ❑ Integrate JavaScript and HTML
- ❑ Create and run simple JavaScript code using different operators

# DOM

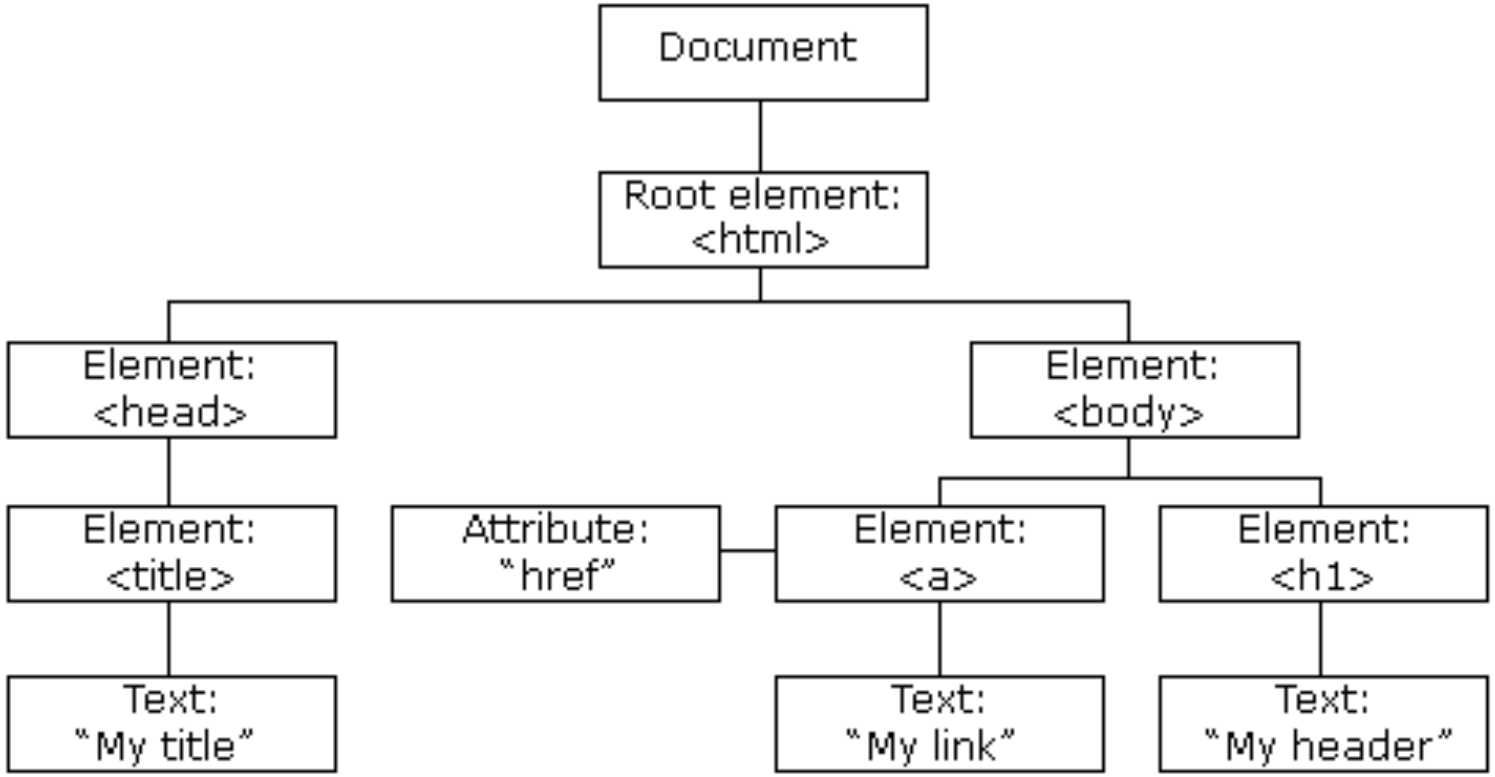
The Document Object Model (DOM) is a programming interface for HTML documents.

- It represents the page so that programs can change the document structure, style and content.
- DOM represents the document as nodes and objects.
- DOM is an object-oriented representation of the webpage, which can be modified with scripting language such as JavaScript.
- All of the properties, methods and events available for manipulating and creating web pages are organised into objects.
- Eg. document object represents the document itself.

# JavaScript and DOM

With HTML DOM, JavaScript can access and change all the elements of an HTML document.

- HTML DOM is constructed as a tree of objects.



# Types of programming

- Broad two categories
  - Client side programming
    - Eg. JavaScript, Vbscript and Jscript
  - Server-side programming
    - Eg. ASP, JSP, PHP, ASP.Net and Java Sevelets

# JavaScript

- For creating interactive webpage
- JavaScript is a client-side scripting language that offers various functionalities such as validating form
- JavaScript is an interpreted language
- JavaScript interpreter interprets the javaScript and thus, embedded in webpages.
- Thus, we define
  - JavaScript is a simple scripting language that is invented specifically for use in web browsers to make websites more dynamic.

# JavaScript

- javaScript was originally named as Livescript
- Javascript can be thought of as a light weight programming language
- Difference between java & javascript

| Java | javaScript |
| --- | --- |
| Created by Sun Microsystem | Created by Netscape |
| is a full computer programming language like C++, suitable for writing complete, large-scale programs | it is very rarely used for anything outside of a browser. |

# Creating simple JavaScript

```
document.write("hello world");
document.write("hello there!!!");
document.write("please help me here");
document.write("I am new here.");
```

# Adding JavaScript in an HTML document

- Must be embedded using <script> and </script> tags
- You can place the <script> tag containing your JavaScript anywhere within your web page but it is preferred way to keep it within the <head> tags.

```
<script ...> JavaScript code </script>
```

- The script tag takes following important attributes:
  - **language:** This attribute specifies what scripting language you are using. Typically, its value will be javascript. Although recent versions of HTML (and XHTML, its successor) have phased out the use of this attribute.
  - **type:** This attribute is what is now recommended to indicate the scripting language in use and its value should be set to "text/javascript".
  - **src:** refers to another file that has scripts

# Adding JavaScript in an HTML document

```
<html>
<body>
<script type="text/javascript">
document.write("Hello World!");
</script>
</body>
</html>
```

# Using an External script file

- Problem:
  - Consider your script that you have created in HTML documents extends to ten and hundreads of line. How will be the readability of your HTML document?
  - What if you want to use same script in several web pages?
- Solution:
  - Use external Javascript **with .js extension**
- After creating the Script, you have to link the HTML document by using the <script> tag
- The external file need only the code and not the script tag.

# Using an External script file

- The following is an example of external javaScript code named ***area.js***

```
var l=5;
var b=6;
var area=(l*b)/2;
document.write(area);
```

- Linking with an HTML document

```
<!DOCTYPE html>
<html>
<head>
<script src="area.js"></script></head>
<body></body>
</html>
```

# Comments in JavaScript

- JavaScript supports both C-style and C++-style comments, Thus:
  - Any text between a // and the end of a line is treated as a comment and is ignored by JavaScript.
  - Any text between the characters /* and */ is treated as a comment. This may span multiple lines.
  - JavaScript also recognizes the HTML comment opening sequence <!--. JavaScript treats this as a single-line comment, just as it does the // comment.
  - The HTML comment closing sequence --> is not recognized by JavaScript so it should be written as //-->

# Case Sensitivity

- JavaScript is a case-sensitive language. This means that language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters.
- So identifiers *Time*, *TIme* and *TIME* will have different meanings in JavaScript.
- **NOTE:** Care should be taken while writing your variable and function names in JavaScript.

# Data types

- What is data types?
- What are the data types in JavaScript?
- JavaScript allows you to work with three primitive data types:
  - Numbers eg. 123, 120.50 etc.
  - Strings of text e.g. "This text string" etc.
  - Boolean e.g. true or false.
- JavaScript also defines two trivial data types, *null* and *undefined*, each of which defines only a single value.
- In addition to these primitive data types, JavaScript supports a composite data type known as *object*.

# Variables

- What is variables?
- All JavaScript **variables** must be **identified** with **unique names**.
  - These unique names are called **identifiers**.
  - Identifiers can be short names (like x and y), or more descriptive names (age, sum, totalVolume).
- The general rules for constructing names for variables (unique identifiers) are:
  - Names can contain letters, digits, underscores, and dollar signs.
  - Names must begin with a letter
  - Names can also begin with $ and _
  - Names are case sensitive (y and Y are different variables)
  - Reserved words (like JavaScript keywords) cannot be used as names

# Variables

- It is unnecessary to declare the type of variables
- You can declare variable with ***var*** and ***let*** keywords
- eg. ***Var*** **d;**
  - Now the variable d is an empty variable
  - To assign value to variable, use = sign as follows
    - Eg. d="google";
  - Or you can assign values at times of declaring the variable
    - Eg. var d="google"
- You can also declare more variable with statement

  eg. Var d=3, e=5, f=6;

# Operators

- for example **4+5=9**
- **4, 5** is an operand and + is an operator
- JavaScript language supports following type of operators.
  - Arithmetic Operators
  - Comparison Operators
  - Logical (or Relational) Operators
  - Assignment Operators
  - Conditional (or ternary) Operators

# Arithmetic Operators

- Assume variable A holds 10 and variable B holds 20 then:

| Operator | Description | Example |
|---|---|---|
| + | Adds two operands | A + B will give 30 |
| - | Subtracts second operand from the first | A - B will give -10 |
| * | Multiply both operands | A * B will give 200 |
| / | Divide numerator by denumerator | B / A will give 2 |
| % | Modulus Operator and remainder of after an integer division | B % A will give 0 |
| ++ | Increment operator, increases integer value by one | A++ will give 11 |
| -- | Decrement operator, decreases integer value by one | A-- will give 9 |

# Arithmetic precedence Operators

- The following table lists JavaScript arithmetic operators, ordered from highest to lowest precedence::

| Operator | Precedence |
| --- | --- |
| ( ) | Expression grouping |
| ++ -- | Increment and decrement |
| * / % | Multiplication, division, and modulo division |
| + - | Addition and subtraction |

# The comparison Operators

- Assume variable A holds 10 and variable B holds 20

| Operator | Description | Example |
|---|---|---|
| == | Checks if the value of two operands are equal or not, if yes then condition becomes true. | (A == B) is not true. |
| != | Checks if the value of two operands are equal or not, if values are not equal then condition becomes true. | (A != B) is true. |
| > | Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true. | (A > B) is not true. |
| < | Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true. | (A < B) is true. |
| >= | Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true. | (A >= B) is not true. |
| <= | Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true. | (A <= B) is true. |

# Logical Operators

- Assume variable A holds 10 and variable B holds 20 then:

| Operator | Description | Example |
|---|---|---|
| && | Called Logical AND operator. If both the operands are non zero then then condition becomes true. | (A && B) is true. |
| \|\| | Called Logical OR Operator. If any of the two operands are non zero then then condition becomes true. | (A \|\| B) is true. |
| ! | Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true then Logical NOT operator will make false. | !(A && B) is false. |

# Assignment Operators

| Operator | Description | Example |
|---|---|---|
| = | Simple assignment operator, Assigns values from right side operands to left side operand | C = A + B will assigne value of A + B into C |
| += | Add AND assignment operator, It adds right operand to the left operand and assign the result to left operand | C += A is equivalent to C = C + A |
| -= | Subtract AND assignment operator, It subtracts right operand from the left operand and assign the result to left operand | C -= A is equivalent to C = C - A |
| *= | Multiply AND assignment operator, It multiplies right operand with the left operand and assign the result to left operand | C *= A is equivalent to C = C * A |
| /= | Divide AND assignment operator, It divides left operand with the right operand and assign the result to left operand | C /= A is equivalent to C = C / A |
| %= | Modulus AND assignment operator, It takes modulus using two operands and assign the result to left operand | C %= A is equivalent to C = C % A |

# Conditional Operators(?:)

- There is an operator called conditional operator. This first evaluates an expression for a true or false value and then execute one of the two given statements depending upon the result of the evaluation. The conditional operator has this syntax

| Operator | Description | Example |
|---|---|---|
| ? : | Conditional Expression | If Condition is true ? Then value X : Otherwise value Y |

# Thank you.