

Web Designing Assignment

Module- 4 (JAVASCRIPT BASIC & DOM)

Q-1) What is JavaScript?

(Ans.):- JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

- JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java.
- JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript.
- The general purpose of the language has been embedded in Netscape, Internet explorer, & other web-browser.
- The ECMA-262 Specification defined a standard version of the core JavaScript language.
 - JavaScript is a light-weight, interpreted programming language.
 - Designed for creating network-centric applications.
 - Complementary to and integrated with Java.
 - Complementary to and integrated with HTML.
 - Open & cross platform.

Q-2) What is the use of isNaN function?

(Ans.):- The isNaN() function is used to check whether a given value is an illegal number or not. It returns true if value is a NaN else returns false.

- The function isNaN determines the argument or the value is a NaN.
- The function returns true if the argument is not a number, otherwise returns false.

In JavaScript, NaN is short for **Not-a-Number**. In JavaScript, NaN is number that is not a legal number. The `Number.isNaN()` method returns true if the value is NaN, and the type is a number.

Syntax:- `isNaN(value)`

Q-3) What is negative Infinity?

(Ans.):- Negative Infinity is the same as the negative value of the global object's Infinity property.

- The negative infinity in JavaScript is a constant value which is used to represent a value which is the lowest available.
- It means that no other number is lesser than this value. It can be generated using self-made function or by an arithmetic operation.
- This value behaves slightly different than mathematical infinity:
- Any positive value, including POSITIVE INFINITY, multiplied by NEGATIVE INFINITY is NEGATIVE INFINITY.
- **NEGATIVE INFINITY** is different from mathematical infinity in the following ways:
 - 1) Negative infinity results in 0 when divided by any other number.
 - 2) When divided by itself or positive infinity, negative infinity return NaN.
 - 3) Negative infinity, when divided by any positive number (apart from positive infinity) is negative infinity.
 - 4) Negative infinity, when divided by any negative number (apart from negative infinity) is negative infinity.
 - 5) If we multiply negative infinity with NaN, we will get NaN as a result.
 - 6) The product of Nan and negative infinity is 0.
 - 7) The product of two negative infinities is always a positive infinity.
 - 8) The product of both positive and negative infinity is always negative infinity.

Syntax:- `Number.NEGATIVE_INFINITY`

Q-4) Which company developed JavaScript?

(Ans.): - The first ever JavaScript was created by **Brendan Eich** at Netscape and became the ECMA-262 standard in 1997.

- After Netscape handed JavaScript over to ECMA, the Mozilla foundation continued to develop JavaScript for the Firefox browser.

Q-5) What are Undeclared & Undefined variables?

(Ans.): - **Undefined** is a variable that has been declared but no values exists and is type of itself 'undefined'.

- It occurs when a variable has been declared but has not been assigned with any value.
- Undefined is not a keyword.

Undeclared variables is a variable that has been declared without 'var' keyword.

- It occurs when we try to access any variable that is not initialized or declared earlier **var** or **const keyword**.
- If we use 'typeof' operator to get the value of an undeclared variable, we will face the runtime error with return value as "**undefined**".
- The Scope of the undeclared variables is always global.

Q-6) Write the code for adding new elements dynamically?

(Ans.): - New elements can be dynamically created in JavaScript with the help of **createElement() method**.

Q-7) What is the difference ViewState and SessionState?

(Ans):- The basic difference between these two is that the **ViewState manage state at the client's end, make state management easy for end-user.**

SessionState manages state at the server's end.

No.	ViewState	SessionState
1)	Maintained at page level only.	Maintained at Session level.
2)	View State can only visible from a single page and not multiple pages.	Session state value availability is across all pages available in a user session.
3)	It will retain values in the event of a postback operation occurring.	In session state, user data remains in the server. Data is available to user until the browser is closed or there is session expiration.
4)	Information is stored on the client's end only.	Information is stored on the server.
5)	It can be used to store information that you wish to access on different web pages.	It can be used to store information that you wish to access from same web pages.
6)	View State values are lost when new page is loded.	Session State can be cleared by programmer or user or in case of timeouts.

Q-8) What is === operator?

(Ans):- In the case of equality comparison, we use === operator, and in the case of inequality comparison, we use !== operator. Return type: boolean. It returns either true or false. The === operator **compares operands and returns true if both operands are of the same data type and have some value, otherwise, it returns false.**

Q-9) How can the style/class of an element be changed?

(Ans):- The class name is used as a selector in HTML which helps to give some value to the element attributes.

The document.getElementById() method is used to return the element in the document with the “id” attribute and the “className” attribute can be used to change/append in class of the element.

Syntax:-

```
document.getElementById('myElement').className = “myclass”;
```

Q-10) How to read and write a file using JavaScript?

(Ans):- **readFile()** and **rs. writeFile()** methods are used to read and write of a file using javascript. The file is read using the **fs.readFile()** function, which is an inbuilt method.

Syntax:- (For reading)

```
fs.readFile(file_name, encoding, callback_function)
```

Syntax:- (For writing)

```
fs.writeFile(file_name, data, options, callback)
```

Q-11) What are all looping structures in Javascript?

(Ans):- The JavaScript loops are used to iterate the piece of coding using **for**, **while**, **do while**, or **for-in** loops. It makes the code compact. It is mostly used in array.

There are four types of loops in JavaScript.

1. for loop
2. while loop
3. do-while loop
4. for-in loop

1) for loop()

➤ **for loop iterates the elements for the fixed number of times.**

Syntax:- for(initialization; condition; increment)

```
{  
  
    code to be executed  
  
}
```

Example:- <! Doctype html>

<html>

<body>

<script>

for(i=1; i<=5; i++)

{

Document.write(i+ "
")

}

</script>

</body>

</html>

Output:- 1

2

3

4

5

2) while loop()

➤ while loop iterates the elements for the infinite number of times.

Syntax:- while(condition)

{

code to be executed

}

Example:- <! Doctype html>

<html>

<body>

<script>

var i=11;

while (i<=15)

{

document.write(i+ "
");

i++;

}

</script>

</body>

</html>

Output:- 11

12

13

14

15

3) do while loop()

- while loop iterates the elements for the infinite number of times like while loop. But, code is executed at least once whether condition is true or false.

Syntax:- do{

Code to be executed

}while (condition);

Example:- <! Doctype html>

<html>

<body>

<script>

var i=21;

do{

document.write(i + "
");

i++;

}while (i<=25);

</script>

</body>

</html>

Output:- 21

22

23

24

25

4) for in loop()

- **For-in loop in JavaScript** is used to iterate over the properties of an object. The for-in loop iterates only over those keys of an object which have their enumerable property set to “true”.

Syntax:-

```
for (let i in obj1) {  
  // Prints all the keys in  
  // obj1 on the console  
  console.log(i);  
}
```

Example:-

```
<script>  
  <script>  
  //declaring a object employee  
  const courses = {firstCourse:'JavaScript',  
                    secondCourse:'React',  
                    thirdCourse:'Angular'};  
  
  let value = '';  
  
  //using for in loop  
  for (let item in courses) {  
    value += courses[item] + '<br>';  
  }  
  document.getElementById("val").innerHTML = value;  
</script>
```

Output:-

Javascript
Angular
React

Q-12) How can you convert the string of any base to an integer in Javascript?

(Ans):- In Javascript, **parseInt()** function (or method) is used to convert the passed in string parameter or value to an integer value itself.

This function returns an integer of base which is specified in second argument of parseInt() function.

❖ Syntax:- parseInt(value, radix)

It accepts a string as a value and converts it to specified radix system (any desired numerical value passed by a user) and returns an integer (corresponding to the passed in numerical radix value).

Example:-

```
<script>
function convertStoI() {
var a = "100";
var b = parseInt(a);
document.write("Integer value is" + b);
var d = parseInt("3 11 43");
document.write("</br>");
document.write('Integer value is ' + d);

}
convertStoI();
</script>
```

Output:-

Integer value is 100

Integer value is 3

Q-13) What is the function of the delete operator?

(Ans):- The **delete** operator **removes a property of an object**. If the property's value is an object and there are no more references to the object, the object held by that property is eventually released automatically.

Syntax:- `delete object.property`
`delete object[property]`


Example:-

```
const Employee = {  
  firstname: 'John',  
  lastname: 'Doe'  
};  
  
console.log(Employee.firstname);  
// expected output: "John"  
  
delete Employee.firstname;  
  
console.log(Employee.firstname);  
// expected output: undefined
```

Output:- “John”

Q-14) What are all the types of Pop up boxes available in JavaScript?

(Ans):- Javascript has three kind of pop up boxes: **Alert box, Confirm box, and Prompt box.**

 **Alert Box:-** It is used when a warning message is needed to be produced.

When the alert box is displayed to the user, the user needs to press ok and proceed.

Syntax:- alert(“your Alert here”)

Example:-

```
<html>
```

```
<body>
```

```
<h2>JavaScript Alert</h2>
```

```
<button onclick="myFunction()">Try it</button>
```

```
<script>
```

```
function myFunction() {
```


```
  alert("I am an alert box!");
```

```
}
```

```
</script>
```

```
</body>
```

```
</html>
```

 **Confirm Box:-** It is type of pop-up box that is used to get authorization or permission from the user. The user has to press the ok or cancel button to proceed.

Syntax:- confirm(“your query here”)

Example:-

```
<html>
<body>


<h2>JavaScript Confirm Box</h2>

<button onclick="myFunction()">Try it</button>

<p id="demo"></p>

<script>
function myFunction() {
  var txt;
  if (confirm("Press a button!")) {
    txt = "You pressed OK!";
  } else {
    txt = "You pressed Cancel!";
  }
  document.getElementById("demo").innerHTML = txt;
}
</script>

</body>
</html>
```

 **Prompt Box:-** It is a type of pop-up box which is used to get the user input for further use. After entering the required details user have to clicks **ok** to proceed next stage else by pressing the cancel button user returns the null value.

Syntax:- prompt(“your prompt here”)

Example:-

```
<html>
<body>
<h2>JavaScript Prompt</h2>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    let text;

    let person = prompt("Please enter your name:", "Harry Potter");
    if (person == null || person == "") {
        text = "User cancelled the prompt.";
    } else {
        text = "Hello " + person + "! How are you today?";
    }

    document.getElementById("demo").innerHTML = text;
}
</script>
</body>
</html>
```

Q-15) What is the use of Void(0)?

(Ans):- Using javascript: you can run code that does not change the current page. This, used with void(0) means, **do nothing - don't reload, don't navigate, do not run any code**. The "Link" word is treated as a link by the browser.

- In a programming language, void means return nothing. “JavaScript: void(0)” is similar to void. javascript: void(0) means **return undefined as a primitive value**. We use this to prevent any negative effects on a webpage when we insert some expression.

Q-16) How can be forced to load another page in JavaScript?

(Ans):- We can use [window.location](#) property inside the *script* tag to forcefully load another page in Javascript. It is a reference to a Location object that is it represents the current location of the document. We can change the URL of a window by accessing it.

Syntax:-

```
<script>
```

```
Window.location = <path/url>
```

```
</script>
```

For example:-

```
<script>
```

```
Window.location = “https://www.amazon.in/”
```

```
</script>
```

Q-17) What are the disadvantages of using innerHTML in Javascript?

(Ans):- Disadvantages of using innerHTML property in JavaScript:

- **The use of innerHTML very slow:-** The process of using innerHTML is much slower as its contents are slowly built, also already parsed contents and elements are also re-parsed which takes times.

- **Content is replaced everywhere:-** Either you add, append, delete or modify contents on a webpage using innerHTML. All contents is replaced, also all the DOM nodes inside that element reparsed and recreated.
- **Can break the document:-** There is no proper validation provided by innerHTML , so any valid HTML code can be used. This may break the document of JavaScript. Even broken HTML can be used, which may lead to unexpected problems.