|  |
| --- |
| **JAVASCRIPT BASIC & DOM** |

**(Q-1) :  What is JavaScript?**

    - JavaScript is a lightweight programming language that web developers commonly

      use to create more dynamic interactions when developing web pages, applications, servers, and or even games.

- Developers generally use JavaScript alongside HTML and CSS The scripting language

works well with CSS in formatting HTML elements. However, JavaScript still maintains user interaction, something that CSS cannot do by itself.

- JavaScript’s implementations within the web, mobile application, and

game development make the scripting language worth learning.

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

    - In JavaScript NaN is short for "Not-a-Number".

    - The isNaN() method returns true if a value is NaN.

    - The isNaN() method converts the value to a number before testing it.

**Example**

- console.log(isNaN(12));           Output : false

  - console.log(isNaN(0 / 0));        Output : true

  - console.log(isNaN(12.3));         Output : false

  - console.log(isNaN("Devanshu"));   Output : true

  - console.log(isNaN("25/05/2023")); Output : true

  - console.log(isNaN(-46));          Output : false

  - console.log(isNaN(NaN));          Output : true

**(Q-3) :  What is negative Infinity?**

- The negative infinity in JavaScript is a constant value that is used to represent a value that is the lowest available. This means that no other number is lesser than this value. It can be generated using a self-made function or by an arithmetic operation.

- JavaScript shows the NEGATIVE\_INFINITY value as –Infinity

**Example :**

let x = Number.NEGATIVE\_INFINITY;

 Output : -Infinity

**Negative infinity is different from mathematical infinity in the following ways:**

- Negative infinity results in -0(different from 0 ) when divided by

      any other number.

    - When divided by itself or positive infinity, negative infinity return NaN

      Negative infinity, when divided by any positive number (apart from positive infinity) is negative infinity.

    - Negative infinity, divided by any negative number

      (apart from negative   infinity) is positive infinity.

    - If we multiply negative infinity with NaN, we will get NaN as a result.

    - The product of 0 and negative infinity is Nan.

    - The product of two negative infinities is always a positive infinity.

    - The product of both positive and negative infinity is always negative infinity.

**(Q-4) : Which company developed JavaScript?**

- JavaScript was invented by Brendan Eich in 1995.

- It was developed for Netscape 2, 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 and undefined variables?**

**Undefined:**

    - It occurs when a variable has been declared but has not been assigned any value.

    - Undefined is not a keyword.

**Example of Undefined :**

    - let a;

      console.log(a)

      Console gives Error like 'a' is undefined.

**Undeclared:**

    - It occurs when we try to access any variable that is not initialized or

      sdeclared earlier using the var or const keyword.

    - If we use ‘typeof’ operator to get the value of an undeclared variable,

      we will face the runtime error with the return value as “undefined”.

    - The scope of the undeclared variables is always global.

**Example of Undeclared:**

- console.log(a)

       //ReferenceError: a is not declared

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

- Javascript is a very important language when it comes to learning how the browser works. Often there are times we would like to add dynamic elements/content to our web pages. This post deals with all of that.

- Creation of new element: New elements can be created in JS by using the createElement() method.

**Syntax:**

-     document.createElement("<tagName>");

- // Where <tagName> can be any HTML

 - // tagName like div, ul, button, etc.

 -  // newDiv element has been created

        For Eg: let newDiv = document.createElement("div");

**(Q-7) : What is the difference between ViewState and SessionState?**

**Differences between ViewState and SessionState:**

| ViewState                            |                               | SessionState

| Maintained at page level only.                 |                        Maintained at session level.

| View state can only be visible from

a single page and not multiple pages. | Session state value availability is across all pages available in a user session.

| 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.

| Information is stored on the client’s end

only.     | Information is stored on the server.

| used to allow the persistence of

page-instance-specific data.             | used for the persistence of user-specific data on the server’s end.

| ViewState values are lost/cleared when

new page is loaded.                 | SessionState can be cleared by programmer or user or in .                                                                 case of timeouts.

**(Q-8) :  What is === operator?**

    - In JavaScript, the === operator is a strict equality comparison operator that returns false

      for values that are not of a similar type.

    - This operator performs type casting for equality.

    - If we compare 2 with “2” using ===, then it will return a false value.

- The == operator is an equality operator that checks whether its two operands are

      the same or not by changing expression from one data type to others.

    - You can use the == operator in order to compare the identity of two operands even

      though they are not of a similar type.

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

    - We can change, add or remove any CSS property from an HTML element on

      the occurrence of any event with the help of JavaScript.

    - There are two common approaches that allow us to achieve this task.

- (1) style.property

    - (2) Changing the class itself

**Approach 1: Changing CSS with the help of the style property:**

**Syntax:**

    document.getElementById("id").style.property = new\_style

**Approach 2: Changing the class itself**

- The classList Property: The classList is a read-only property that returns the

      CSS class names of an element as a DOMTokenList object.

**Syntax:**

    document.getElementById("id").classList

- You can use the below-mentioned methods to add classes, remove classes, and toggle between different classes respectively.

- The add() method: It adds one or more classes.

    - The remove() method: It removes one or more classes.

    - The toggle() method: If the class does not exist it adds it and returns true.

      It removes the class and returns false. The second boolean argument forces the class to be added or removed.

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

**(1) Write operation on a file**

    - After the File System file is imported then, the writeFile() operation is called.

    - The writeFile() method is used to write into the file in JavaScript.

**Syntax**

    writeFile(path,inputData,callBackFunction)

**The writeFile() function accepts three parameters −**

**Path**

    - The first parameter is the path of the file or the name of the file into which

      the input data is to be written.

- If there is a file already, then the contents in the file are deleted and

      the input which is given by the user will get updated or if the file is not present,

then the file with that will be created in the given path and the input information is written into it.

**inputData**

    - The second parameter is the input data which contains the data to be written

      in the file that is opened.

**callBackFuntion**

    - The third parameter is the function which is the call back function which takes

      the error as the parameter and shows the fault if the write operation fails.

**(2) Reading from the file**

    - After the File System module is imported, the reading of the file in JavaScript

      can be done by using the readFile() function.

**Syntax**

     readFile(path, format, callBackFunc)

**The readFile() function accepts three parameters including one optional parameter.**

**Path**

    - The first parameter is the path of the test file from which the contents are to read.

    - If the current location or directory is the same directory where the file which is

      to be opened and read is located then, only the file name has to be given.

**Format**

    - The second parameter is the optional parameter which is the format of the text file.

    - The format can be ASCII, utf-8 etc.

**CallBackFunc**

    - The third parameter is the call back function which takes the error as the parameter

      and displays the fault is any raised due to the error.

**(Q-11) : What are all the looping structures in JavaScript?**

    - JavaScript mainly provides three ways for executing the loops.

    - While all the ways provide similar basic functionality, they differ in their syntax and condition-checking time.

**while loop**

    - A while loop is a control flow statement that allows code to be executed

      repeatedly based on a given Boolean condition.

    - The while loop can be thought of as a repeating if statement.

**Syntax**

while (condition) {

        loop statements...

    }

**for loop**

    - for loop provides a concise way of writing the loop structure.

    - Unlike while loop, a for statement consumes the initialization, condition,

      and increment/decrement in one line thereby providing a shorter, easy-to-debug structure of looping.

**Syntax**

for (initialization; condition; increment/decrement) {

        statement(s)

    }

**do-while**

    - The do-while loop is similar to the while loop with the only difference

      is that it checks for the condition after executing the statements,

      and therefore is an example of an Exit Control Loop.

**Syntax**

do {

        Statements..

    }

    while (condition);

**(Q-12) : How can you convert the string of any base to an integer in JavaScript?**

- The parseInt function available in JavaScript

parseInt(string, radix);

Where, the paramters are the following −

**String**

    - The value to parse. If this argument is not a string, then it is converted to one using the ToString method. Leading whitespace in this argument is ignored.

**Radix**

    - An integer between 2 and 36 that represents the radix (the base in mathematical numeral systems) of the string.

So we can pass the string and the radix and convert any numbner with base from 2 to 36 to integer using this method.

**Example**

    - console.log(parseInt("100", 10))  Output : 100

    - console.log(parseInt("10", 8))    Output : 8

    - console.log(parseInt("101", 2))   Output : 5

    - console.log(parseInt("2FF3", 16)) Output : 12275

    - console.log(parseInt("ZZ", 36))   Output : 1295

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

    - The JavaScript pop(), shift(), or splice() methods are available to delete an element from an array.  - But because of the key-value pair in an object, deleting is more complicated.

    - Note that, the delete operator only works on objects and not on variables or functions.

**Syntax:**

delete object

    or

    delete object.property

    or

    delete object['property']

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

- In Javascript, popup boxes are used to display the message or notification to the user.

    - There are three types of pop-up boxes in JavaScript namely Alert Box, Confirm Box and Prompt Box.

**(1) 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");

**(2) Confirm Box:**

- It is a 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("Confirm This...");

**(3) 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 click ok to proceed next stage

      else by pressing the cancel button user returns the null value.

**Syntax:**

prompt("your Prompt here");

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

- The void keyword in JavaScript is a unary operator that evaluates

      its operand and always returns the undefined primitive value.

    - Its primary purpose is to create an expression that performs

      an action without returning a value, or to create a shortcut for returning undefined from a function.

**Example :**

<a href="#" onclick="doSomething(); void 0;">Click me</a>

- we're using void to prevent the browser from navigating to a new page when the link is clicked.

    - The doSomething() function is called, and then void 0 is used to create an expression that evaluates to undefined.

**(Q-16) : How can a page be forced to load another page in JavaScript?**

- In JavaScript, we can use window.location object to force a page to load another page.

    - We can use the location object to set the URL of a new page.

    - There are different ways – window.location.href property, window.location.assign() and window.location.replace() methods,to set the URL of a new page using the location object.

    - We will discuss each of the property and methods in detail in this tutorial.

**Window.location.replace**

- The first way is to use the window.location.href property

- This property contains information about the current URL of the page, and it can be used to redirect the user to a new page.

**Syntax**

window.location.href = "new\_url";

    - The user will be immediately redirected to the specified URL (new\_url).

- To redirect the user after a specified amount of time has passed,

      we may also specify the setTimout function which allows the user to redirect

      to the source URL after the time specified in the function.

setTimeout(function() {

       window.location.href = "https://www.youtube.com";

    }, 2000);

**(Q-17) :  What are the disadvantages of using innerHTML in JavaScript?**

Disadvantages of using innerHTML property in JavaScript:

**The use of innerHTML very slow:**

    - The process of using innerHTML is much slower as its contents as slowly built,

      also already parsed contents and elements are also re-parsed which takes time.

**Preserves event handlers attached to any DOM elements:**

    - The event handlers do not get attached to the new elements created by setting innerHTML automatically.

    - To do so one has to keep track of the event handlers and attach it to new elements manually.

    - This may cause a memory leak on some browsers.

**### 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 are reparsed and recreated.

**### Appending to innerHTML is not supported:**

    - Usually, += is used for appending in JavaScript. But on appending to an Html tag using innerHTML, the whole tag is re-parsed.