MODULE:4(JavaScript Basic & DOM)

1. **What is JavaScript?**

**=>**JavaScript is a programming language that is used to create interactive websites and web applications. It is a high-level, interpreted language that is commonly used for client-side scripting, which means that it runs in the web browser of the user.

2) **What is the use of isNaN function?**

**=>**The isNaN() function is used to determine whether a value is NaN (Not-a-Number) or not. It returns true if the value is NaN, and false if it is not NaN.

**3) What is negative Infinity?**

**=>**Negative infinity is a value that represents the smallest possible number in JavaScript. It is a constant that is returned when a mathematical operation results in a value that is too small to be represented by a normal number.

4) **Which company developed JavaScript?**

**=>**JavaScript was developed by Netscape Communications Corporation, which is now known as Mozilla Corporation.

5) **What are undeclared and undefined variables?**

**=>**Undefined variable means a variable has been declared but it does not have a value.

undeclared variable means that the variable does not exist in the program at all.

6) **Write the code for adding new elements dynamically?**

**=>**javascript

// get the parent element

var parent = document.getElementById("parent-element");

// create a new element

var newElement = document.createElement("div");

// add some content to the new element

newElement.innerHTML = "This is a new element!";

// add the new element to the parent element

parent.appendChild(newElement);

7) **What is the difference 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. |

**8) What is ==operator?**

**=>**The `==` operator is a comparison operator in JavaScript that compares two values for equality. It returns `true` if the values are equal and `false` otherwise.

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

## =>1: Changing CSS with the help of the style property:

**Syntax:**

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

**Example:** In this example, we have built a PAN number validator. First, we will take the input value and match it with a regex pattern. If it matches then using JavaScript add an inline style on the <p> tag. Otherwise, add a different style on the <p> tag.

<!DOCTYPE html>

<**html** lang="en">

<head>

<meta charset=”UTF-8”>

<meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>

<meta name=”viewport” content=”width-device-width, initial-scale=1.0”>

<title>document</title>

</head>

<**body**>

    <**h1** style="color: green;">

        GeeksforGeeks

    </**h1**>

    <**h2**>

        How can the style/class of

        an element be changed?

    </**h2**>

    <**b**>Validate Pan Number</**b**>

    <**input** type="text" id="pan" />

    <**p**></**p**>

    <**button** id="submit">Validate</**button**>

    <**script**>

        const btn = document.getElementById("submit");

        btn.addEventListener("click", function () {

            const pan = document.getElementById("pan").value;

            const para = document.querySelector("p");

            let regex = /([A-Z]){5}([0-9]){4}([A-Z]){1}$/;

            if (regex.test(pan.toUpperCase())) {

                para.innerHTML = "Hurrey It's correct";

                // Inline style

                para.style.color = "green";

            } else {

                para.innerHTML = "OOps It's wrong!";

                // Inline style

                para.style.color = "red";

            }

        });

    </**script**>

</**body**>

</**html**>

**2. The className Property:**This property is used to set the current class of the element to the specified class.

**Syntax:**

document.getElementById("id").className = class

**Example:**

* HTML

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| <!DOCTYPE html>  <**html** lang="en">    <**head**>  <meta charset=”UTF-8”>  <meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>  <meta name=”viewport” content=”width-device-width, initial-scale=1.0”>  <title>document</title>      <**style**>          .colorBlue {              color: blue;          }            .colorRed {              color: red;          }      </**style**>  </**head**>    <**body**>      <**h1** style="color: green;">          GeeksforGeeks      </**h1**>        <**h2**>          How can the style/class of          an element be changed?      </**h2**>        <**h3**>className Example</**h3**>        <**p** class="colorBlue">          GeeksforGeeks is a computer science portal          for geeks.This platform has been designed          for every geek wishing to expand their          knowledge, share their knowledge and is          ready to grab their dream job. GFG have          millions of articles, live as well          as online courses, thousands of tutorials          and much more just for the geek inside you.      </**p**>        <**button** id="submit">Change Color</**button**>        <**script**>          const btn = document.getElementById("submit");          const para = document.querySelector("p");            btn.addEventListener("click", function () {              para.className = "colorRed";          });      </**script**>  </**body**>    </**html**>  10) **How to read and write a file using JavaScript?**  **=>**In a web browser environment, JavaScript does not have direct access to the file system for security reasons. However, you can use the File API to read and write files using JavaScript.  Ex:- javascript  // get the file input element  var fileInput = document.getElementById("my-file-input");  // get the file object  var file = fileInput.files[0];  // create a new file reader  var reader = new FileReader();  // set the onload event handler  reader.onload = function(e) {  // get the file contents  var contents = e.target.result;  // do something with the contents  };  // read the file  reader.readAsText(file);  11) **What are all the looping structures in JavaScript?**  **=>**There are three main types of loops in JavaScript: `for`, `while`, and `do-while`.  The `for` loop is typically used when you know how many times you want to iterate through a loop. Here's an example:  javascript  for (var i = 0; i < 10; i++) {  console.log(i);  }  In this example, the loop will iterate 10 times and log the value of `i` to the console on each iteration.  The `while` loop is typically used when you don't know how many times you want to iterate through a loop. Here's an example:  javascript  var i = 0;  while (i < 10) {  console.log(i);  i++;  }  12)  **How can you convert the string of any base to an integer in JavaScript?**  **=>**You can use the `parseInt()` function to convert a string of any base to an integer in JavaScript. The `parseInt()` function takes two arguments: the string to be parsed and the base of the number system used in the string.  **Example:** In this example, we would be passing the string value in a method (which is explicitly declared for ease purpose) and further that string value is passed inside the parseInt() method which then further converts that string value in the corresponding integer value.  JavaScript   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | <script>      let stringConversion = (string\_value) => {        console.log("Initial Type: " + **typeof** string\_value);        let integer\_value = parseInt(string\_value);        console.log("Final Type: " + **typeof** integer\_value);        console.log(integer\_value);      };        stringConversion("512000");      stringConversion("126410");      stringConversion("0x8975");  </script>  **13) What is the function of the delete operator?**  **=>**The `delete` operator is used to delete a property from an object. Here's an example:  javascript  var myObj = { foo: "bar", baz: "qux" };  delete myObj.foo;  console.log(myObj); // Output: { baz: "qux" }  In this example, the `delete` operator is used to remove the `foo` property from the `myObj` object.  Note that the `delete` operator only removes the property from the object, not the value of the property. If you want to remove the value as well, you can set it to `undefined`:  javascript  var myObj = { foo: "bar", baz: "qux" };  delete myObj.foo;  myObj.baz = undefined;  console.log(myObj); // Output: { baz: undefined }  14) **What are all the types of Pop up boxes available in JavaScript?**  **=> There are three types of pop-up boxes available in JavaScript:**  1. `alert()` - displays a message in a dialog box and waits for the user to click "OK".  2. `confirm()` - displays a message in a dialog box and waits for the user to click either "OK" or "Cancel". Returns a boolean value indicating the user's choice.  3. `prompt()` - displays a message in a dialog box and waits for the user to enter some text. Returns the text entered by the user as a string.  Example:  1)alert Box:-  **Example:**   |  | | --- | | <!DOCTYPE html>  <**html**>    <**head**>  <meta charset=”UTF-8”>  <meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>  <meta name=”viewport” content=”width-device-width, initial-scale=1.0”>      <**title**>Pop-up Box type | Alert Box</**title**>      <**style**>          h1{              color:green;          }      </**style**>  </**head**>    <**body**>      <**center**>            <**h1**>GeeksforGeeks</**h1**>            <**h3**>Alert Box</**h3**>          <**button** onclick="geekAlert()">              Click here for alert box          </**button**>            <!-- Alert box function -->          <**script**>              function geekAlert() {                  alert("An Online Computer Science"                              + "Portal for Geeks");              }          </**script**>      </**center**>    </**body**>    </**html**> |  1. confirm box;-   **Example:**   |  |  | | --- | --- | | **xample:**   |  | | --- | | <!DOCTYPE html>  <**html**>    <**head**>  <meta charset=”UTF-8”>  <meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>  <meta name=”viewport” content=”width-device-width, initial-scale=1.0”>      <**title**>          Pop-up Box type | Confirm Box      </**title**>      <**style**>          h1 {              color: green;          }      </**style**>  </**head**>    <**body**>      <**center**>            <**h1**>GeeksforGeeks</**h1**>            <**h3**>Confirm Box</**h3**>            <**button** onclick="geekConfirm()">              Click here for Confirm box          </**button**>            <**p** id="geek"></**p**>              <!-- Confirm box function -->          <**script**>              function geekConfirm() {                  var x;                  if (confirm("Press a button!") == true) {                      x = "OK pressed!";                  } else {                      x = "Cancel!";                  }                  document.getElementById("geek").innerHTML = x;              }          </**script**>      </**center**>  </**body**> | |  1. prompt box:-   Example:-  <!DOCTYPE html>  <**html**>    <**head**>  <meta charset=”UTF-8”>  <meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>  <meta name=”viewport” content=”width-device-width, initial-scale=1.0”>      <**title**>          Pop-up Box type | Prompt Box      </**title**>      <**style**>          h1 {              color: green;          }      </**style**>  </**head**>    <**body**>        <**center**>            <**h1**>GeeksforGeeks</**h1**>            <**h3**>Prompt Box</**h3**>            <**input** type="button" onclick="geekPrompt();"          value="Click here for Prompt box"/>            <!-- Prompt box function -->          <**script**>              function geekPrompt() {                  var x = prompt("Enter your mail here : ");                  document.write("Your ID : " + x);              }          </**script**>      </**center**>    </**body**>   </**html**>  **15) What is the use of Void (0)?**  **=> Using “javascript:void(0);” in anchor tag:**Writing “javascript:void(0);” in anchor tag can prevent the page to reload and JavaScript functions can be called on single or double clicks easily. **Example:**  html   |  |  |  | | --- | --- | --- | | <!DOCTYPE html>  <**html lang=”en”**>    <**head**>  <meta charset=”UTF-8”>  <meta Http-equiv=”X-UA-Compatible” content=”IE=edge”>  <meta name=”viewport” content=”width-device-width, initial-scale=1.0”>  <**title**>JavaScript:void(0)</**title**>  </**head**>    <**body**>      <**center**>          <**h1** style="color:green">GeeksforGeeks</**h1**>          <**h3**>JavaScript:void(0)</**h3**>          <**a** href="javascript:void(0);"             ondblclick="alert('Welcome to Geeks for Geeks')">  Double click on me </**a**>      </**center**>  </**body**>    </**html**>  **16)  How can a page be forced to load another page in JavaScript?** => Ans. **Step 1:** Create a file named ***index.html***. Add a heading and two buttons to it. One button forcefully loads a page with a live URL and the other button loads a local HTML page. In the *<script>* tag we have two functions, one loads gfg home page, and the second loads a local HTML page using ***window.location*** property.   * index.html  |  | | --- | | <!DOCTYPE html>  <**html** lang="en">    <**head**>      <**meta** charset="UTF-8">      <**meta** http-equiv="X-UA-Compatible"          content="IE=edge">      <**meta** name="viewport" content=          "width=device-width, initial-scale=1.0">  </**head**>    <**body**>      <**h3**>This is the original page</**h3**>      <**br**>        <**button** onclick="force\_load\_gfg()">          Force Load GFG Page      </**button**>      <**br**><**br**>        <**button** onclick="force\_load\_local()">          Force Load Local HTML page      </**button**>        <**script**>          function force\_load\_gfg() {              window.location =                  "<https://www.geeksforgeeks.org/>"          }            function force\_load\_local() {              window.location =                  "F:/gfg/PageRedirect/newPage.html"          }      </**script**>  </**body**>    </**html**> |   **Step 2:** Create a file named ***newPage.html***. This is the local HTML page that would be loaded by Javascript.   * newPage.html  |  | | --- | | <!DOCTYPE html>  <**html** lang="en">    <**head**>      <**meta** charset="UTF-8">      <**meta** http-equiv="X-UA-Compatible"          content="IE=edge">      <**meta** name="viewport" content=          "width=device-width, initial-scale=1.0">      <**title**> New Page </**title**>  </**head**>    <**body**>      <**h3**>This is the new loaded page</**h3**>  </**body**>    </**html**> |   17)  **What are the disadvantages of using innerHTML in JavaScript?**  **=>**Disadvantages of inner HTML  event handler attached to any DOM element are preserved.  replacement is done everywhere.  breack the document.  used for cross-site scripting. | | |