|  |
| --- |
| **Question:** What is JavaScript? |
| **Answer:**  JavaScript is a scripting or programming language that allows you to implement complex features on web pages — every time a web page does more than just sit there and display static information for you to look at — displaying timely content updates, interactive maps, animated 2D/3D graphics, scrolling video jukeboxes, etc. — It is the third layer of the layer cake of standard web technologies, two of which (HTML and CSS) we have covered in much more detail in other parts of the Learning Area. |
| **Question:** What is the use of isNaN function? |
| **Answer:**  The JavaScript isNaN() Function is used to check whether a given value is an illegal number or not. It returns true if the value is a NaN else returns false. It is different from the Number.isNaN() Method. |
| **Question:** What is negative Infinity? |
| **Answer:**  NEGATIVE\_INFINITY is a special numeric value that is returned when an arithmetic operation or mathematical function generates a negative value greater than the largest representable number in JavaScript (i.e., more negative than -Number. MAX\_VALUE). JavaScript displays the NEGATIVE\_INFINITY value as –Infinity. |
| **Question:** Which company developed JavaScript? |
| **Answer:**  JavaScript was created at Netscape Communications by Brendan Eich in 1995. Netscape and Eich designed JavaScript as a scripting language for use with the company's flagship web browser, Netscape Navigator. |
| **Question:** What are undeclared and undefined variables? |
| **Answer:**  **Undefined**: It occurs when a variable has been declared but has not been assigned any value. Undefined is not a keyword.  **Undeclared**: It occurs when we try to access any variable that is not initialized or declared earlier using the var or const keyword. |
| **Question:** Write the code for adding new elements dynamically. |
| **Answer:**  <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Add new element dynamically</title>  </head>  <body>      <h1 style="color:green;">          Add Element Dynamically      </h1>      <!-- Form to add item -->      <form action="#">            <!-- Type of Element -->          <label for="type">              Add Element Type          </label>            <input type="text" id="type" placeholder="Like: div, h1, li...." value="li" />          <br /><br />            <!-- Text/Value for the element --->          <label for="value">              Add Element Value          </label>            <input type="text" id="value" placeholder="Like: Hello GeeksForGeeks" value="CHILD 2" />          <br /><br />            <!-- Submit the Form -->          <button type="button" onClick="addItem()">              Add          </button>      </form>        <!-- Parent tag where we add              item as child -->      <ol id="parent">          <li>List Item 1</li>      </ol>        <script>          // Define the addItem() function          // to be called through onclick          function addItem() {                // Get type of element from form              let type = document.                  getElementById("type").value;                // Get the text/value for the tag              // from the form              let value = document.                  getElementById("value").value;                // createElement() is used for              // creating a new element              type                  = document.createElement(type);                // Use value as textnode in this example              type.appendChild(                  document.createTextNode(value));                // Append as child to the parent              // tag i.e. ol              document.getElementById(                  "parent").appendChild(type);          }      </script>    </body>  </html> |
| **Question:** What is the difference between View State and Session State? |
| **Answer:**  The differences between View State and Session State are as follows.   |  |  | | --- | --- | | **View state** | **Session state** | | State management is on the client side. | State management is on the server side. | | Data is accessible within the same web page. | Data is accessible from other web pages of the website. | | Loss of information occurs when different web page is loaded. | Loss of information occurs due to timeout. | | Data is stored in a hidden field of the same web page. | Data is stored in sessions and cookies. | | It's less secure. | It's more secure. | |
| **Question:** What is === operator? |
| **Answer:**  === (Triple equals) is a strict equality comparison operator in JavaScript, which returns false for the values which 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. |
| **Question:** How can the style/class of an element be changed? |
| **Answer:**  **Approach 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 name="viewport" content="width=\, initial-scale=1.0">      <title>Change CSS</title>  </head>      <body>              <h1 style="color: green;">          Change CSS      </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>  **Approach 2**: Changing the class itself – We can use two properties that can be used to manipulate the classes.  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.  Example: In this example, we are using the above-explained approach.  <!DOCTYPE html>      <meta charset="UTF-8">      <meta name="viewport" content="width=\, initial-scale=1.0">      <title>Change Class</title>  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=\, initial-scale=1.0">      <title>Change Class</title>      <style>          .hide {              display: none;          }          .blueColor {              color: blue;          }      </style>  </head>  <body>      <h1 style="color: green;">          Change Class      </h1>        <h2>          How can the style/class of          an element be changed?      </h2>        <h3>Hide and Show the Para</h3>        <p>          This platform has been designed          for every one 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="hide">Hide</button>      <button id="show">Show</button>      <button id="color">Change Color</button>      <script>          const btn\_hide = document.getElementById("hide");          const btn\_show = document.getElementById("show");          const btn\_color = document.getElementById("color");          const para = document.querySelector("p");          btn\_hide.addEventListener("click", function () {              para.classList.add("hide");          });          btn\_show.addEventListener("click", function () {              para.classList.remove("hide");          });          btn\_color.addEventListener("click", function () {              para.classList.toggle("blueColor");          });      </script>  </body>  </html> |
| **Question:** How to read and write a file using JavaScript? |
| **Answer:**  The read and write operations in a file can be done by using some commands. But the module which is required to perform these operations is to be imported. The required module is ‘fs’ which is called as File System module in JavaScript.  **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. The syntax of this method is as follows –  **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.  **Code snippet**:      <script>          const fs = require('fs')          let fInput = "You are reading the content from Tutorials Point"          fs.writeFile('tp.txt', fInput, (err) => {          if (err) throw err;          else{console.log("The file is updated with the given data")}          })      </script>  **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.  **Code snippet**:          const fs = require('fs')          fs.readFile('tp.txt', (err, inputD) => {          if (err) throw err;              console.log(inputD.toString());          }) |
| **Question:** What are all the looping structures in JavaScript? |
| **Answer:**  In JavaScript, there are 7 types of loops that can be used to execute a block of code repeatedly: for, for-in, for-of, while, do-while, forEach, and map. Each loop has its own specific use case and syntax, and choosing the right one depends on the problem you’re trying to solve.  A **for loop** is used when you know how many times you need to repeat a certain block of code. It takes three statements.   * Initialization statement * Condition Statement * Increment statement.   A **while loop** is used when you don’t know how many times you need to repeat a block of code, but you know the condition that will end the loop.  A **do-while loop** is similar to a while loop, but the block of code is executed at least once, even if the condition is false.  A **for-in loop** is used to loop through the properties of an object.  A **for-of loop** is used to loop through the values of an iterable object (such as an array.  A **for Each loop** is a method on arrays that executes a function for each element in the array.  A **map loop** is a method on arrays that creates a new array by executing a function on each element in the array. |
| **Question:** How can you convert the string of any base to an integer in JavaScript? |
| **Answer:**  To convert a string to an integer parseInt(), Number(), and Unary operator(+) function is used in javascript. parseInt() function returns Nan( not a number) when the string doesn't contain number. If a string with a number is sent, then only that number will be returned as the output. This function won't accept spaces. |
| **Question:** What is the function of the delete operator? |
| **Answer:**  The delete operator removes a given property from an object. On successful deletion, it will return true , else false will be returned. |
| **Question:** What are all the types of Pop up boxes available in JavaScript? |
| **Answer:**  There are three types of pop-up boxes in JavaScript namely Alert Box, Confirm Box and Prompt Box. |
| **Question:** What is the use of Void (0)? |
| **Answer:**  “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. |
| **Question:** How can a page be forced to load another page in JavaScript? |
| **Answer:**  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. |
| **Question:** What are the disadvantages of using innerHTML in JavaScript? |
| **Answer:**  It is very slow because as inner HTML already parses the content even we have to parse the content again so that's why it takes time. When we have used the event handlers then the event handlers are not automatically attached to the new elements created by innerHTML. |