

**Module (JAVASCRIPT BASIC & DOM) – 4**

1. What is JavaScript?

JavaScript is a versatile programming language commonly used in web development to enhance the user experiences on the internet.

It is primarily utilized for implementing functionalities within our web applications. JavaScript is often hailed as a the "language of the web", is the most important part of modern web development.

1. What is the use of isNaN function?

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 NaN else returns false.

1. 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.

Syntax: Number.NEGATIVE\_INFINITY;

1. Which company developed JavaScript?

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.

1. What are undeclared and undefined variables?

Simply undeclared is a variable that is not declared.

And undefined is one that is declared without any value

Forexample:

let x;

console.log(x)

output: undefined

Because the value of x is not defined

console.log(y);

output:Undeclare

Because y is not declared or defined

We can say every undeclared variable is undefined

but every undefined variable is not undeclared.

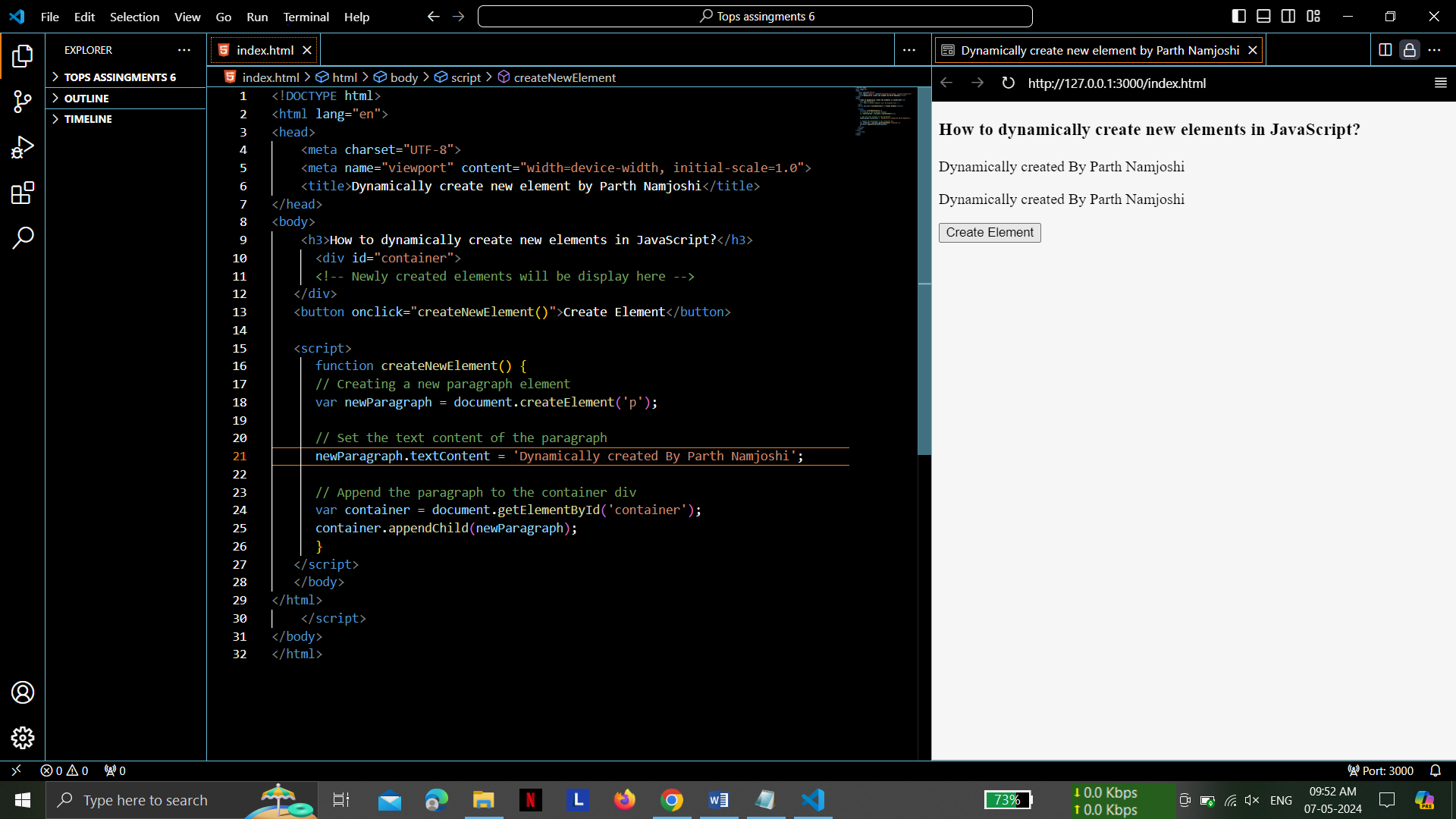
It is possible to declare variable and still it remains undefined, until we give it some value.

1. Write the code for adding new elements dynamically?

Dynamic element creation in JavaScript allows to generate new HTML elements on the get-go.

Whether we need to add content to a web page based on user interactions or dynamically generate elements for a specific task, it can help enhance the flexibility and interactivity of web applications.

For example



1. What is the difference between ViewState and SessionState?

ViewState and SessionState are used for client-side state management and server-side

state management respectively.

The basic difference between these two is that the

ViewState is to manage state at the client’s end, making state management easy for end-

User.

While SessionState manages state at the server’s end, making it easy to manage

content from this end too.

**ViewState:** It is maintained at only one level that is page-level. Changes made on a single page is not visible on other pages.

Information that is gathered in view state is stored for the clients only and cannot be transferred to any other place.

When view state is used, the values posted of a particular page persist in the browser that the client is using and post back only when the entire operation is done.

The data of the previous page is no longer available when another page is loaded.

Also, Data is not secure in this case because it is exposed to clients.

Encryption can be used for data security.

**SessionState:** It is maintained at session-level and data can be accessed across all pages in the web application.

The information is stored within the server and can be accessed by any person that has access to the server where the information is stored.

It can be used to store information that you wish to access on different web pages.

1. What is === operator?

=== Is Strict Equality Operator.

It used to compare two operands and return true if both the value and type of operands are the same.

Since type conversion is not done, so even if the value stored in operands is the same but their type is different the operation will return false.

1. How can the style/class of an element be changed?

To change the style or class of an element in JavaScript,

We have a few different options

depending on what exactly we want to modify.

We can change both the style

properties and the class of an HTML element using JavaScript:

***Changing Style Property***

We can directly manipulate the style property of an element to change specific CSS

styles.

By Style Property Directly:

element.style.color = "red";

        element.style.fontSize = "20px";

Using setAttribute:

element.setAttribute("style", "color: red; font-size: 20px;");

Adding or removing Classes

// Adding a class

element.classList.add("new-class");

// Removing a class

element.classList.remove("old-class");

// Toggling a class (adds if not present, removes if present)

element.classList.toggle("active");

***Changing Class***

We can directly manipulate the class of an element using the className or classList property.

Using `className`

// Set the class attribute directly

element.className = "new-class";

Using `classList`

// Adding a class

element.classList.add("new-class");

// Removing a class

element.classList.remove("old-class");

// Toggling a class (adds if not present, removes if present)

element.classList.toggle("active");

1. How to read and write a file using JavaScript?

The fs.readFile() and rs.writeFile() methods are used to read and write of a file using javascript.

*Using fs.readFile() function*

The file is read using the function, which is an inbuilt method.

This technique reads the full file into memory and stores it in a buffer.

Syntax:

fs.readFile( file\_name, encoding, callback\_function )

filename:

It contains the filename to be read, or the whole path if the file is saved elsewhere.

encoding:

It stores the file’s encoding. ‘utf8’ is the default setting.

callback function:

This is a function that is invoked after the file has been read.

err:

If there was an error.

data:

The file’s content.

*Using rs.writeFile() function*

The fs.writeFile() function is used to write data to a file in an asynchronous manner. If the file already exists, it will be replaced.

Syntax:

fs.writeFile( file\_name, data, options, callback )

1. What are all the looping structures in JavaScript?

* JavaScript Loops are powerful tools for performing repetitive tasks efficiently.
* Loops in JavaScript execute a block of code again and again while the condition is true.
* Loops are integral to maintaining the readability of our code and reducing pressure on the system by minimizing the number of lines needed to accomplish a given task.

This directly enhances the quality of the code, improves runtime efficiency, and saves

considerable time

that would otherwise be spent writing extensive code.

For example,

suppose we want to print “Hello Parth Namjoshi!” 5 times. This can be done using JS Loop easily. In Loop, the statement needs to be written only once and the loop will be executed 5 times as shown below:

for (let i = 0; i < 5; i++) {

    console.log("Hello Parth Namjoshi!");

}

Output:

Hello Parth Namjoshi!

Hello Parth Namjoshi!

Hello Parth Namjoshi!

Hello Parth Namjoshi!

Hello Parth Namjoshi!

There are primarily three types commonly used.

1. For loop

The most fundamental and widely used loop is 'for' loop. The "for" loop in JavaScript allows us to execute a block of code repeatedly until a specified condition is met. It's particularly useful when we know how many times you want to iterate.

A for loop is constituted with four parts: the initialization, the condition, increment/decrement and the code which has to be executed if the conditions are met true.

Syntax:

for (initialization; condition; increment/decrement) {

    // code block to be executed

}

1. While loop

The "while" loop in JavaScript provides another method to execute a block of code repeatedly based on a specific condition.

The primary difference between the 'for' loop and the 'while' loop lies in their syntax, also we use 'while' loop when the number of iterations is not predetermined or when we want to keep looping until a specific condition is met.

Syntax:

while (condition) {

    // code to be executed

}

1. Do-while loop

The 'do-while' loop in JavaScript is similar to the 'while' loop but, with one key difference: the condition is checked after the loop body executes. Due to this property if we use 'do-while' loop, the code inside the loop will execute at least once, regardless of the initial condition.

Syntax:

do {

    // code to be executed

} while (condition);

JavaScript encompasses two additional loops within its domain.

1. for/in loop

JS for/in loop 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 variable\_name in object\_name) {

    // Statement

}

1. for/of loop:

JS for/of loop is used to iterate the iterable objects for example – array, object, set and map. It directly iterate the value of the given iterable object and has more concise syntax than for loop.

for(let variable\_name of  object\_name) {

    // Statement

}

1. How can you convert the string of any base to an integer in JavaScript?

Converting a string to an integer in JavaScript means transforming a string that represents a numeric value into an actual integer data type.

This is useful when we need to perform arithmetic operations or comparisons on numeric values stored as strings.

We have different methods to convert the string to integer

* parseInt() Method in JavaScript :

The parseInt() method accepts the string and radix parameter and converts it into an integer.

Syntax: parseInt( Value, radix )

* Number() Method in JavaScript

The number() method is used to convert primitive data type to a number, if it is not convertible it returns NAN.

Syntax: Number( value )

* Unary Operator in JavaScript

The Unary operator(+) is used to convert a string, boolean, and non-string to a number.

* Math.floor() Method in JavaScript

The Math.floor() method is used to convert a string into number.

1. What is the function of the delete operator?

The JavaScript delete operator deletes/ removes a property from an object. It removes the property as well as value of the property from the object. It works only with the objects not with the variables or functions.

1. What are all the types of Pop up boxes available in JavaScript?

JavaScript provides various popup boxes to notify, warn, or to get input from the user. Popup boxes prevent the user from accessing other aspects of a program until the popup is closed, so they should not be overused.

There are three different kinds of popup methods used in JavaScript.

* Alert box

An alert dialog box is mostly used to inform or alert the user by displaying some messages in a small dialogue box.

* Confirm box

A confirmation box is used to let the user make a choice. When Javascript pops up a confirm box, the user will have to click either "OK" or "Cancel" to proceed to the next step.

* Prompt box

Javascript Prompt Box can be used when we want to get some user input. When Javascript displays a prompt box, the user will see a popup box with an input field and buttons "OK" or "Cancel" to proceed after entering an input value.

1. What is the use of Void (0)?

In English, void means nothing. 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.

For example,

in the case of URL hyperlinks. Hyperlinks open by reloading the page when the user clicks on the link. When you need to run some other code in such cases, you can use javascript: void(0).

1. How can a page be forced to load another page in JavaScript?

In JavaScript, we can force a page to load another page by using the window.location object.

There are a few methods to achieve this. To force a page to load another page in JavaScript, we have multiple approaches:

Below are the approaches used to force a page to load another page in JavaScript:

* Using window.location.replace:

The replace function is used to navigate to a new URL without adding a new record to the history.

* Using window.location.assign Property

The assign function is similar to the href property as it is also used to navigate to a new URL.

The assign method, however, does not show the current location, it is only used to go to a new location.

Unlike the replace method, the assign method adds a new record to history (so that when the user clicks the “Back” button, he/she can return to the current page).

1. What are the disadvantages of using innerHTML 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.
* **Old content replaced issue:** The old content is replaced even if object.innerHTML = object.innerHTML + ‘html’ is used instead of object.innerHTML += ‘html’. There is no way of appending without reparsing the whole innerHTML. Therefore, working with innerHTML becomes very slow. String concatenation just does not scale when dynamic DOM elements need to be created as the plus’ and quote openings and closings becomes difficult to track.
* **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.