Sunday
8-Feb-2026

SHREE

Date: / / Page No.

2nd JavaScript Class

Variable

var - Deprecate
(isko use mat karo)

use let & Const

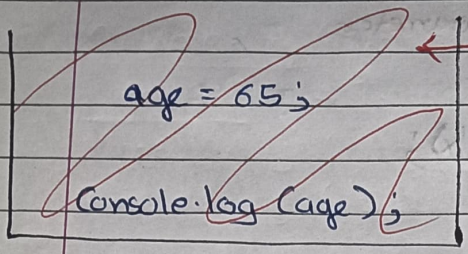
interview question
is let is hoisted?
- yes + explaination
~~because It is tepo~~
Because it is temporal dead zone (TDZ)

Temporal Dead Zone (TDZ)

- A Temporal Dead Zone in Javascript is the period from the start of a block scope ({}) until a let or Const variable is actually declared and initialized, during which the variable exists but is inaccessible, causing a ReferenceError if accessed. Unlike var variables, which are hoisted but undefined

let variable

Example,



mp	cp
age = undefined. → if you use let in TDZ then It's gona through error	

```
let age = 24;
Console.log (age);

age = 45;
Console.log (age);
```

Const variable

you can't assign const variable to any one.
you can only assign for one time.

Examples

```
Const age = 24;
```

- It has similar behaviour how let variable It also makes TDZ.

Functions

- function - a reusable set of instructions
- A block of code designed to perform a specific task.

Syntax

```
function functionName () {
    // code stats here
}
```

- add input called as parameters

ex,

```
function greetUser (x) {
    Console.log ('Hey ' x);
}
greetUser ('Piyush');
```

function name — Parameter — x value

(you can also add more than one parameter.)

- String interpulation (``)

you can add variable under the backtick

ex,

```
function greetUser (x, y, z) {
    Console.log (`hey, ${x} and ${y}`);
}
greetUser ('Dipali', 'Sharma', '.');
```

function names — Parameters — interpulate the variable

- Expressions

ex,

```
function add (num1, num2) {
    Const result = num1 + num2;
    return result;
}
Const r = add (2, 5)
for (let i = 0; i < 5; i++) {
    Console.log (`value of i is ${i}`);
}
```

return is the last statement of the function

Output =

value of i is 0

value of i is 1

- function inside function

ex,

```
function cartoon () {
    function cartoonInsideCartoon() {
        return 'Naruto';
    }
    return cartoonInsideCartoon;
}
```

function incide Function

return the function

```
const anime = cartoon();
const x = anime()
console.log (x);
```

Type of anime is function

called the function

assain the value

- you can called the function incide the function.
- function can return any think like string, number, or can even function.

- ultemate way to make function

ex,

```
let cartoon = function () {
    Console.log ('Anime');
};
cartoon();
```

ex,

```
age = 45;
Console.log ('Value of age is', age, 'Is Allowed?',
                    isAllowedToVote (age));
```

```
var age = 24;

function isAllowedToVote (age) {
    return age >= 18;
}
```

mp	cp
age (variable)	~~undefined~~ 45
function isAllowedToVote (age) { return age >= 18; }	

output :-

Value of age is 45 . Is Allowed? True

- If define let & Const insted of var
  It gives you ReferenceError (TDZ)

```
var isAllowedToVote = function (age) {
        return age >= 18;
    };
```

- undefined ko aap called nahi kar sakte.

Arrow function (=>)

- you can skip the function keyword, the return keyword and the curly brackets.

ex,

function name — Parameter — return

```
Const isAllowedToVote = (age) => age >= 18;

Console.log (isAllowedToVote (23));
```

```
Const isUserAllowedToOpenBankAccount =
                (age, minBalance) => {
    return age >= 18 && minBalance >= 5000;
}

Console.log (isUserAllowedToOpenBankAccount (23, 6000));
```

Two Parameters

- If you have more than one statement then you have to add parane thesis.
- there is no need to add paranthesis if you have one parameter
- If you use paranthesis the you have to use return keyword too.

Data Structure

- memory mai data ko ek perticular structure mei store karna

SHREE

Date: / / Page No.

Dat structure of Javascript

- Array ([ ])

- an array is a single object used to store an ordered collection of multiple values under a single variable name.
- Arrays are dynamic, zero-indexed (starts with 0)
- you can use different type of data type like string, number, Boolean

ex,

```
Const fruits = ['apple', 'cheeku', true, 'aada', 1, 🍎,
                'santra', 'kela'];
```

(0 1 2 3 4 5 6 7)

- Console.log (fruits[4]);

output

aada

- Console.log (fruits[10]);

output

undefined (no error for out of brand)

- you can also print using

```
Console.log (fruits);
```

output

apple	cheeku
true	aada
🍎	santra
1	kela

- add (end me kuch add)

fruits.push ('Kiwi');

- Print the length

Console.log (fruits.length);

- Check the value in array

Console.log (fruits.includes ('aadu')

output

true.

- Returns a copy of a section of an array.

Const firstElement = fruits.slice (2, 5)

index 2 se 5 tak sare element de do.

assigment

Queeles and stack Implement karna hai using array.

- Console.log ({firstElement});
- fruits.unshift ('1', '2', '3');
- Console.log (fruits.indexOf());

High Order functions

-A function that takes another function as a parameter

ex,

```
function meraPyaraFunction (udharKaFunction) {
    return udharKaFunction () + 40;
}
```

call the function

Point toward cartoon

```
function cartoon () {
    return 10;
}
```

simple function

```
Console.log (meraPyaraFunction (cartoon));
```

cartoon is Parameter

called the function

then 10 + 40

Output

50

Ex,

```
function meraPyaraFunction (udharKaFunction) {
    return udharKaFunction () + 40;
}
function meraEkAurFunction () {
    return 100;
}
console.log (meraPyaraFunction (meraEkAurFunction));
```

call the function

100 + 40

Parameter as Function

called the function

total = 140

forEach

ex,

```
Const fruits = ['apple', 'cheeku', 'true', 'aadu', '1',
                'santra', '0', 'kela'];

fruits.forEach((x4z) => console.log(`--> ${x4z}`));
```

Build in foreach calling function for every element.

string interpolation

```
function forEach (bataoKyaKarnaHai) {
    for (let i=0; i < fruits.length; i++) {
      bataoKyaKarnaHai (fruits[i]);
    }
    }
```

Map

makes new array

ex,

```
Const nums = [1,2,3,4,5,6]
Const result = nums.map((e) => e*2);
Console.log (result);
```

output

[2,4,6,4,10,12]

```
Const nums = [1,2,3,4,5,6];

                    call the Function
Const result = map((e) => e*3);
Console.log (result);

function map (fn) {          → e => e*3
    Const result= [];

for (let i=0; i < nums.length; i++) {
    Const currentElement = nums [i];
    Const num = fn (currentElement);
    result.push (num);
    }

    return result;
}
```

Output:-

[3, 6, 9, 14, 15, 18];

OR

```
Const nums = [1,2,3,4,5,6];

Const result = nums.map ((e) => e*3 );
Console.log (result);

function map (fn){
    Const rusult= [ ];
  }
```