30 Days Javascript Challenge

Chai aur Code

Day 1 - Variables and Datatypes

Day-1 - Variables and Datatypes.

Page No
Date 13/7/24 1

11/11

1) Variables - This are the Containers for storing data.

JS Variable Can be declared in 4ways

- Automatically.
- Using var
- Using let
- Using Const.

★ Automatically.

```
x = 5                    Console.log (z)
y = 6                    //output: 11

z = x+y
```

* It is considee good programming to alway declare a variable before use.

★ Using Var.

```
Var x= 5;
Var y= 6;                Console.log (z)

Var z= x+y               //output: 11
```

* The Var Keyword was used in all JS code from 1995 to 2015

* The var Keyword should only be used in code Written for old Browsers.

* let and Const Keywords were added to Js in 2015

* Using let:

```
let x = 5;
let y = 6;

let z = x+y;
```

```
console.log (z)
//output : 11
```

* Using Const:

The values in const are Constant value and Cannot be change or rewrite it.

```
Const x = 5;
Const y = 6

Const z = x +y;
```

```
console.log (z)
//output : 11.
```

* Mixed Example:

```
Const price1 = 100;

Const price2 = 200;

let total = price1 + price2;

console.log (total)

  // output : 300;
```

* When to use Var, let or Const?

1) Always use Const If the value Should not be Changed.

2) Alwayse use Const If Type Should not be changed.

3) Only use let If you Can't use Const

4) Only use Var if you Must Support Old browsers.

* Javascript Identifiers.

1) All Js Variable must be identified with unique nomes.

This unique names are Called Identifiers

Identifiers Can be Short names like (x, y) Or more Descriptive names like (age, Value totalSum)

* Rules For Constructing names For Variable.

1) name Can Contain letters, digit, underscore and dollar sign.

2) names must begin with a letter.

3) Names can also begin with ($ and -)

4) Names are case sensitive (y and Y) are two diffrent variables.

5) Reseeved Keywords like, let, const var cannot be used as variable.

* Js Identifiers (variables) are case sensitive

* Decleoring a Js Variable.

Createing a variable in Js is called 'Declaring' a variable.

let CarName;

↳ After Declearation the variable has no value in it. (It is undefined)

To assign value to CarName we use equal sign:

ex - let CarName = "BMW"

* It is good programming practis to always declor all the variable at the beginning of a script.

★ One statement many Variables.

1) you can declare many Variables in <u>one</u> Statement

~~Stati~~ Starting the statement with let and seprate it by comma's

ex-

```
let person = "Joh Doe";
      age = 23;
      City = " pune"
```

★ Value = undefined.

A variable declare without value will have the value <u>undefined</u>.

ex-

```
    let CarName
```

The variable CarName will have value undefined.

★ Re-Decleaning JS Variable

If you redeclare a JS Variable · decleard already with var it will not lose it's value.

The variable CarName will still have value "volvo"

ex-

```
Var CarName = "volvo"
Var CarName;
```

* You Cannot re-declare a variable with let and Const.

This will not work ↓

```
let CarName = "volvo"

let CarName;
```

* This will throw error.

★ Javascript let

1) let Keyword was Introduced in Es6 2015

2) Variable declared with let has Block scope.
   ↳ They cannot be access Outside the block.

```
ex- {
        let CarName = "volvo";          Block
        Console.log (CarName)           Scope
        Output: Volvo
    }
    Console.log (CarName)
  ! error : is not defined.
```

3) Variable declared with let cannot be Redeclared in Same Scope.

```
ex {
      let city= "Pune"
      let city          ⟶ error
    }                     already declared
```

* Javascript Const.

1) Const Keyword was also Introduced in ES6 - 2015.

2) Variable defined with const cannot be redeclared.

3) Variable defined with Const cannot be Reassigned.

4) Variables Defined with Const have Block scope.

★ Datatypes

JS has 8 datatypes

1) String
2) Number
3) Bigint
4) Boolean
5) Undefined
6) Null
7) Symbol
8) Object

★ The Object Datatype

The Object Datatype contains both built in objects and User defined objects.

1) Built-in objects.

Built in Objects are pre-made by Javascript, ready to use.

ex- Objects, Array, dates, map sets, promises, Math

2) User - defined Objects.

User defined Object are objects made by you to suit your specific needs.

```
ex    Class Car {  model : "256DY";
                   Color : "Red"
```

Datatypes examples.

1) Number.

```
let length = 16
let weight = 7.5;
```

2) //string.

```
let Color = "Yellow"
let lastNome = "Johnson"
```

3) Booleans.

```
let x = ~~F~~ true;
let y = false.
```

4) //object.

```
Const person = {
                firstNome: "Bhavesh";
                lastNome: "Nandowe"
              }
```

5) Array object.

```
Const ~~person~~ cars = ["Volvo", "BMW", "Audi"]
```

6) Date object.

```
Const date = new Date ("2022-3-25")
```

* A Javascript variable can hold any type of data.

* When adding number and a string JS will treat them as a String.

1) ex -

```
let x = 16 + "Volvo"

Console.log (x)

// Output : 16Volvo
```

2) ex -

```
let x = "Volvo" + 16

Console.log (x)

// Output : Volvo16
```

* Js evaluates expression from left to right. Diffrent Sequence can provide diffrent results.

ex-1
```
let x = 16 + 4 + "volvo"

Console.log (x)

//output 20volvo
```

↳ In this example JS treats 16 and 4 as numbers until it reach volvo

But

```
let x = "volvo" + 16+4

console.log (x)

//output Volvo164
```

↳ In this exampue as first opeeand is String all opeeand are treated as String

★ Javascript Types are Dynamic

Javascript has dynamic type. This mean Same variablu can be use to hold diffrent Data types.

```
let x;          // now x is undefined.
x = 5           // now x is a number
x= "John"       // now x is a string.
```

★ Javascript String.

A string (or a text string) is a series of Characters like "Joh Doe"

String are written with quotes single or Doublu.

```
ex-1)  let carName1= "volvo"
       let CarName2= 'BMW'
```

★ Javascript Numbers.

All Js Numbers are written with or without decimals.

1) With decimals.

```
let x1 = 34.00;
```

2) Without Decimal.

```
let x2 = 34;
```

★ Javascript Bigint.

All Js numbers are stored in a 64-bit floating point format.

Js Bigint is a new datatype (ES2020) that can be use to store integer value that are too big to be represented by a normal Js numbe.

ex-

```
let x = BigInt("123456789012345678901234")
```

★ Javascript Booleans.

Booleans can only have two value true or false.

Booleans are often use in conditional Testing.

```
let x = 5;
let y = 6;

let z = 6

(x == y)    //return Falce.

(y == z)    // return True.
```

★ Javascript Array.

Js array are written with squase brackets.

Array items are seprated by Comma

↳ The below code declare (Created) an array called Cars containing items (carnames)

ex -

```
Const cars = ["volvo", "BMW", "Audi"]
```

Array Indexes are zero based which mean the first item is [0] and second is [1] and so on.

★ Javascript objects.

Js objects are written with Curly Braces { }

Object properties are written as

Name: Value

Key: value pairs. Seprated by Comma.

ex -

```
const person = { firstName = "John",
                 lastName = "Doe",
                 age: 50,
                 city: "pune"
               }
```

The object person above has 4 properties firstName, lastName, age, city.

★ Typeof operator.

you can use Js typeof operator. to find type of Javascript Variable.

The typeof operator return type of a variable or an expression.

ex

```
1) typeof ""          //Returns "string"
   typeof "John"      //Returns "string"
   type of "John Doe" //Returns String"
```

2)
```
typeof 0          // Returns "number"
typeof 314        // Returns "number"
typeof 3.14       // Returns "number"
typeof(3+4)       // Returns "number"
```

★ Undefined.

In Javascript, a variable without value has the value undefined
The type is also undefined.

ex -

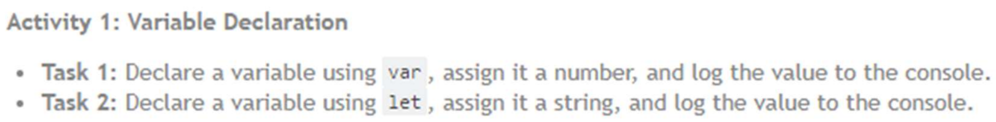
```
let car;     //value is undefined.
```

★ Empty Values

```
let value = " ".
```

↳ The value is " "
and type of is string.

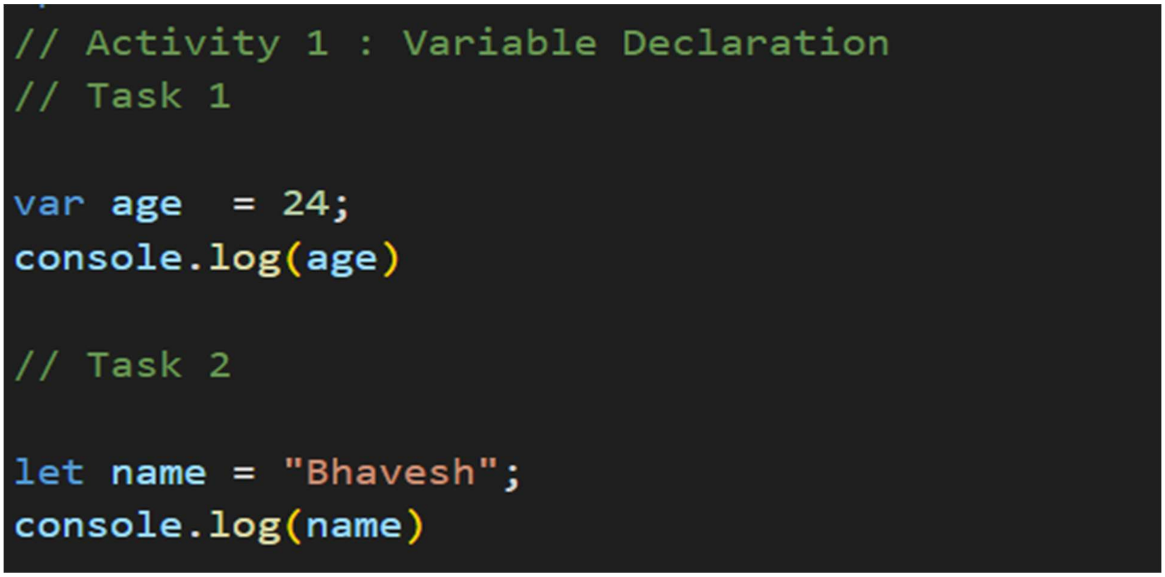
Activity 1



Activity 1: Variable Declaration

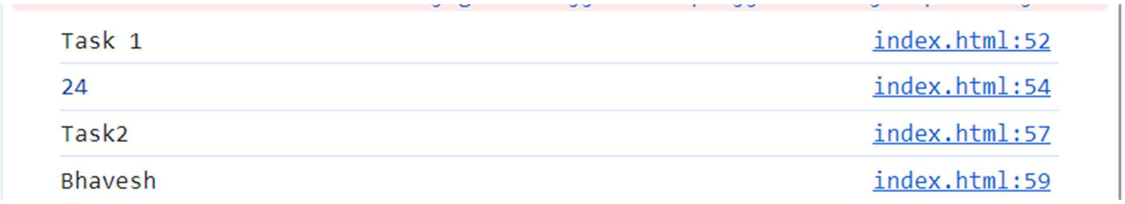
- **Task 1:** Declare a variable using `var`, assign it a number, and log the value to the console.
- **Task 2:** Declare a variable using `let`, assign it a string, and log the value to the console.

Code



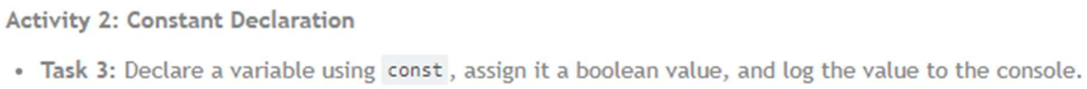
```
// Activity 1 : Variable Declaration
// Task 1

var age  = 24;
console.log(age)

// Task 2

let name = "Bhavesh";
console.log(name)
```

Output



```
Task 1                                             index.html:52
24                                                 index.html:54
Task2                                              index.html:57
Bhavesh                                            index.html:59
```

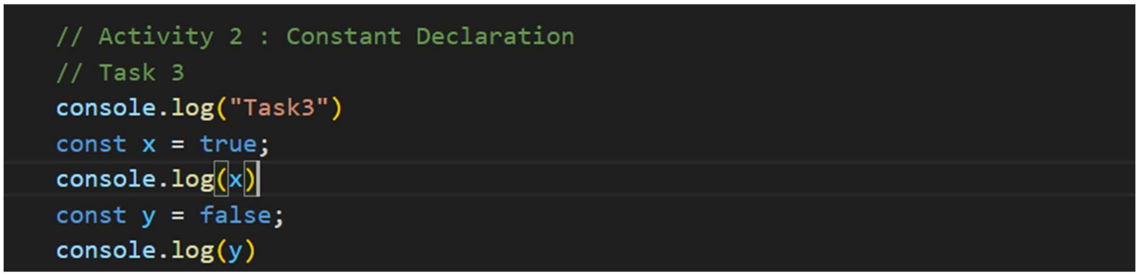
Activity 2



Activity 2: Constant Declaration

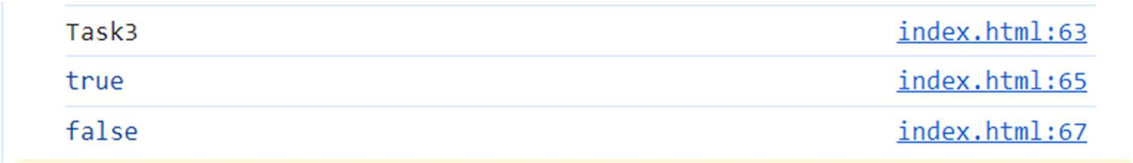
- **Task 3:** Declare a variable using `const`, assign it a boolean value, and log the value to the console.

Code



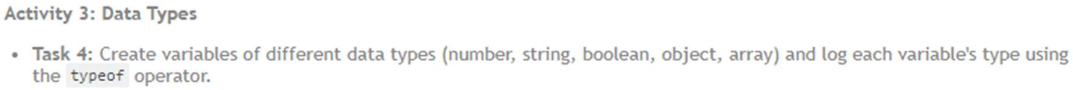
```
// Activity 2 : Constant Declaration
// Task 3
console.log("Task3")
const x = true;
console.log(x)
const y = false;
console.log(y)
```

Output



```
Task3                                                   index.html:63
true                                                    index.html:65
false                                                   index.html:67
```

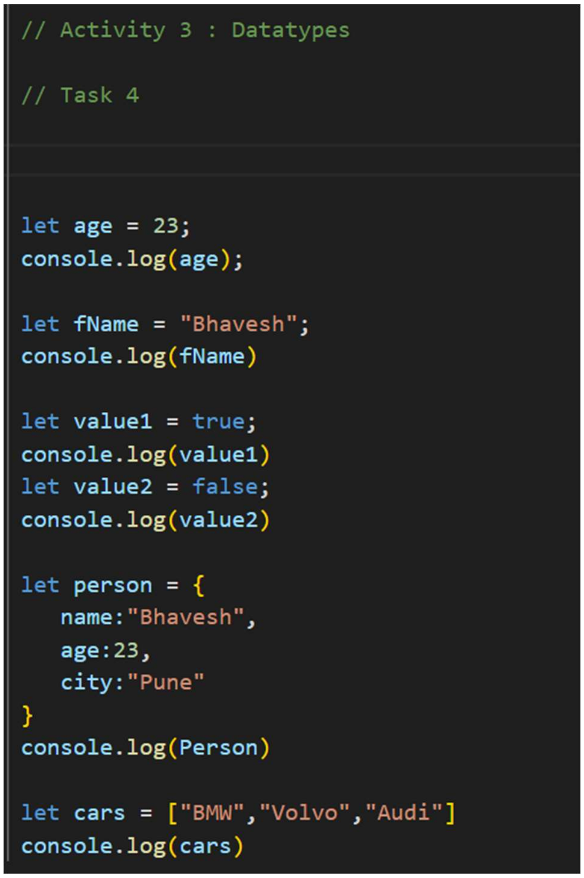
Activity 3



Activity 3: Data Types

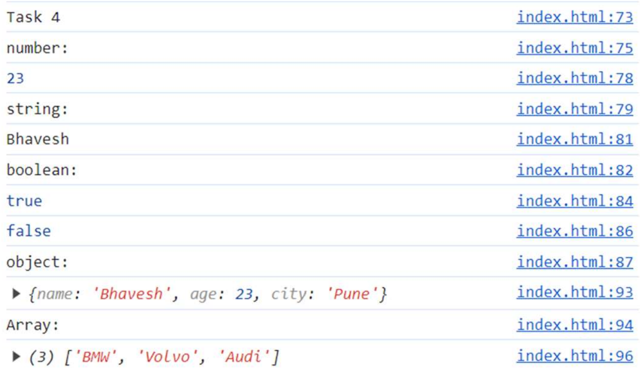
- **Task 4:** Create variables of different data types (number, string, boolean, object, array) and log each variable's type using the `typeof` operator.

Code



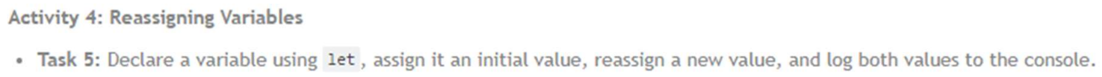
```
// Activity 3 : Datatypes

// Task 4



let age = 23;
console.log(age);

let fName = "Bhavesh";
console.log(fName)

let value1 = true;
console.log(value1)
let value2 = false;
console.log(value2)

let person = {
   name:"Bhavesh",
   age:23,
   city:"Pune"
}
console.log(Person)

let cars = ["BMW","Volvo","Audi"]
console.log(cars)
```

Output



```
Task 4                                                  index.html:73
number:                                                 index.html:75
23                                                      index.html:78
string:                                                 index.html:79
Bhavesh                                                 index.html:81
boolean:                                                index.html:82
true                                                    index.html:84
false                                                   index.html:86
object:                                                 index.html:87
 ▶ {name: 'Bhavesh', age: 23, city: 'Pune'}             index.html:93
Array:                                                  index.html:94
 ▶ (3) ['BMW', 'Volvo', 'Audi']                         index.html:96
```

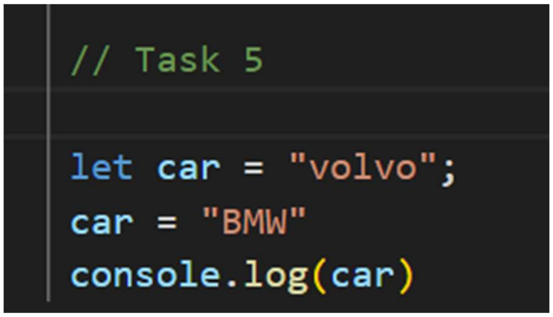
Activity 4



Activity 4: Reassigning Variables

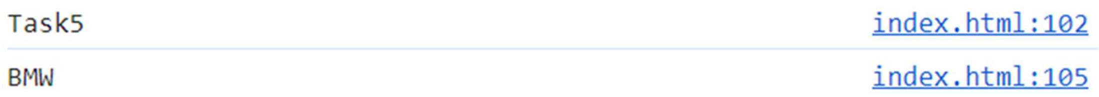
- **Task 5:** Declare a variable using `let`, assign it an initial value, reassign a new value, and log both values to the console.

Code



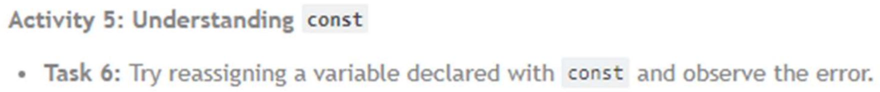
```
// Task 5

let car = "volvo";
car = "BMW"
console.log(car)
```

Output



```
Task5                                              index.html:102
BMW                                                index.html:105
```

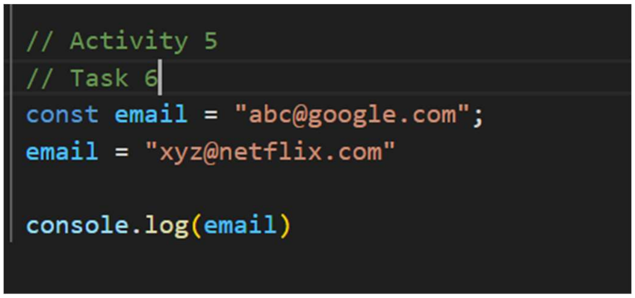
Activity 5



Activity 5: Understanding `const`

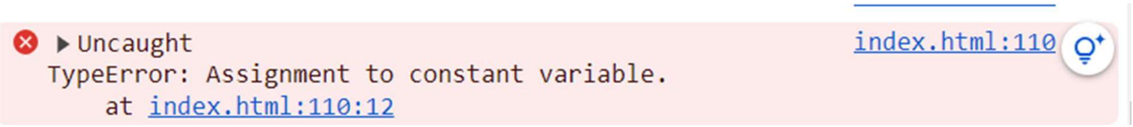
- **Task 6:** Try reassigning a variable declared with `const` and observe the error.

Code



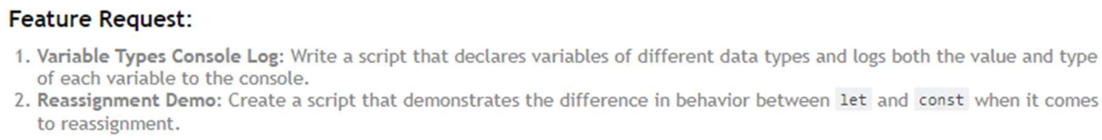
```
// Activity 5
// Task 6
const email = "abc@google.com";
email = "xyz@netflix.com"

console.log(email)
```

Output



```
▶ Uncaught                                         index.html:110
  TypeError: Assignment to constant variable.
      at index.html:110:12
```

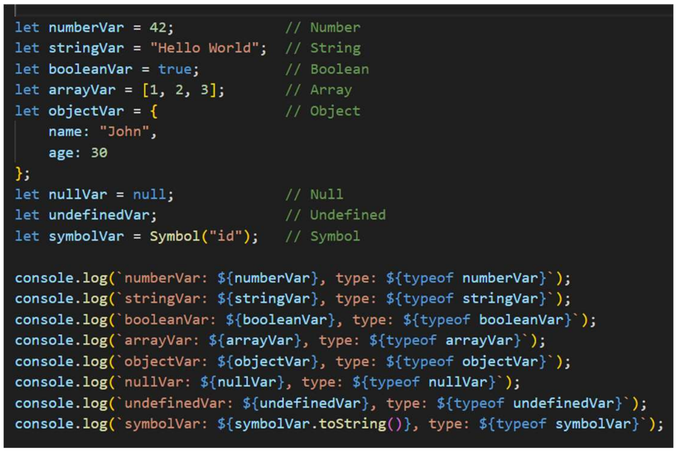
Feature Request 1)



Feature Request:

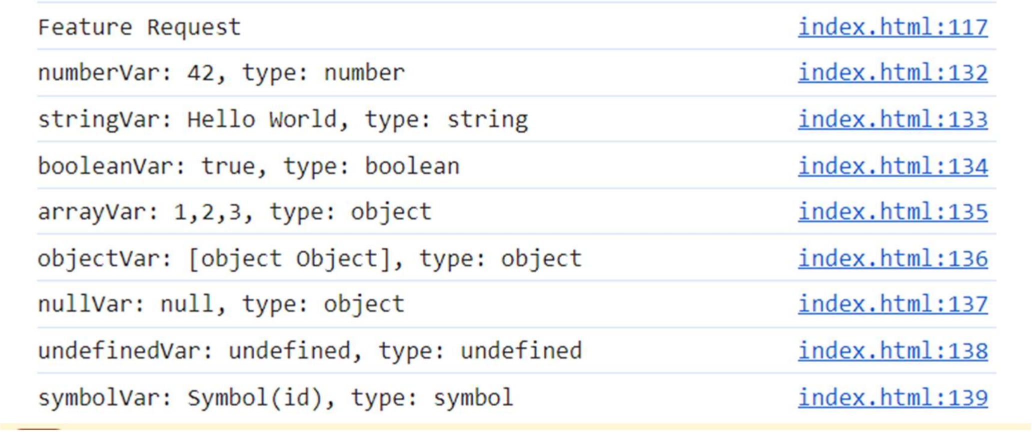
1. **Variable Types Console Log:** Write a script that declares variables of different data types and logs both the value and type of each variable to the console.
2. **Reassignment Demo:** Create a script that demonstrates the difference in behavior between `let` and `const` when it comes to reassignment.

Code



```
let numberVar = 42;             // Number
let stringVar = "Hello World";  // String
let booleanVar = true;          // Boolean
let arrayVar = [1, 2, 3];       // Array
let objectVar = {               // Object
    name: "John",
    age: 30
};
let nullVar = null;             // Null
let undefinedVar;               // Undefined
let symbolVar = Symbol("id");   // Symbol

console.log(`numberVar: ${numberVar}, type: ${typeof numberVar}`);
console.log(`stringVar: ${stringVar}, type: ${typeof stringVar}`);
console.log(`booleanVar: ${booleanVar}, type: ${typeof booleanVar}`);
console.log(`arrayVar: ${arrayVar}, type: ${typeof arrayVar}`);
console.log(`objectVar: ${objectVar}, type: ${typeof objectVar}`);
console.log(`nullVar: ${nullVar}, type: ${typeof nullVar}`);
console.log(`undefinedVar: ${undefinedVar}, type: ${typeof undefinedVar}`);
console.log(`symbolVar: ${symbolVar.toString()}, type: ${typeof symbolVar}`);
```

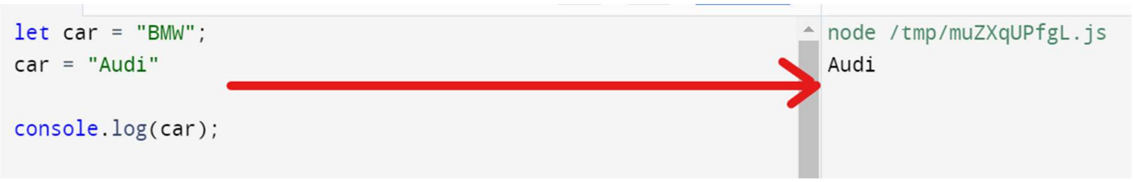
Output



```
Feature Request                                   index.html:117
numberVar: 42, type: number                       index.html:132
stringVar: Hello World, type: string              index.html:133
booleanVar: true, type: boolean                   index.html:134
arrayVar: 1,2,3, type: object                     index.html:135
objectVar: [object Object], type: object          index.html:136
nullVar: null, type: object                       index.html:137
undefinedVar: undefined, type: undefined          index.html:138
symbolVar: Symbol(id), type: symbol               index.html:139
```

Feature Request 2)

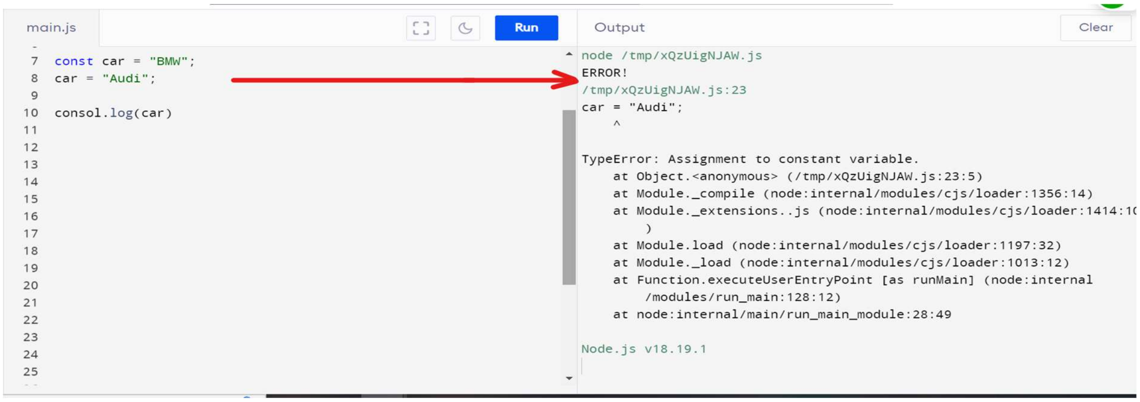
Code



```
let car = "BMW";
car = "Audi"

console.log(car);
```

```
node /tmp/muZXqUPfgL.js
Audi
```

Vs



```
const car = "BMW";
car = "Audi";

consol.log(car)
```

```
node /tmp/xQzUigNJAW.js
ERROR!
/tmp/xQzUigNJAW.js:23
car = "Audi";
    ^

TypeError: Assignment to constant variable.
    at Object.<anonymous> (/tmp/xQzUigNJAW.js:23:5)
    at Module._compile (node:internal/modules/cjs/loader:1356:14)
    at Module._extensions..js (node:internal/modules/cjs/loader:1414:1
        )
    at Module.load (node:internal/modules/cjs/loader:1197:32)
    at Module._load (node:internal/modules/cjs/loader:1013:12)
    at Function.executeUserEntryPoint [as runMain] (node:internal
        /modules/run_main:128:12)
    at node:internal/main/run_main_module:28:49

Node.js v18.19.1
```

By-

Bhavesh Nandave

X

LinkedIn