Java Script

- Java Script is a client side web programming language.
- Java script can update and change both HTML and CSS.
- Java script can calculate, manipulate and validate data.
- It is used to capture user actions on a web page.

Printing in JS→

```
console.log("Hitesh")
```

Variables and constant→

```
const accountId = 145623
let accountEmail = "vasu@gmail.com"
var accountPassword = 1445
accountCity = "Jaipur"
```

- we declare constant using const keyword and once declared it can't be changed
- There are three ways to declare variables→
→ by using let keyword
→ by using var keyword
→ not using any keyword

prefer not to use var keyword because of issue in block scope and functional scope.

☆ NOTE→ Always use let keyword

- now it will be very difficult to print everything using console.log () and we can print them easily using console.table () which will display in form of table.

console.table([accountId, accountEmail, accountPassword, accountCity])

<u>Output</u>

(index)	Values
0	145623
1	'vasu@gmail.com'
2	1445
3	'Jaipur'

- In Javascript it is optional to end statement with ";".

- If we didn't assign any value and just declare the variable then it will show undefined on printing.

let accountState;
console.log(accountState)

<u>Output</u> → undefined

☆ "use strict";
//add this at top of your JS file, this will treat all JS code as newer version

• // alert(3+3)
/* if we use this alert in web console (in inspect) then it will pop up a alert message but we are using node.js not browser, hence it will give an error */
→ For more info refer to Java Script mdn.

Null and undefined
Null is a stand alone value, now in simple words it means, that the variable is not undefined but intentionally empty.

let state = null

undefined means that variable has not been assigned any value.
Consider them same as data type.

Symbol
These datatype basically means unique. It will be discussed more when we will be learning React components.

Type of
This will tell what is the data type of given variable or input. There are two ways:
(i) console.log (typeof "vasu")

Output →
string

(ii) console.log (typeof variable_name)

Special case

console.log (typeof null)

Output →
object

- null is considered as of type object

Conversion in Javascript

```
let score = "33"
let scoreInNumber = Number(score)
console.log (scoreInNumber)
```

- similarly
→ String()
→ Boolean()

→ now there are some special cases
- in Number() →

(i) "vasu" => value will be NaN but type shown is number
(ii) "33" => 33
(iii) null => value will be 0 and type is number
(iv) true => 1 and False => 0
(v) "33abc" => value will be NaN but type shown is number

(ii) Boolean() →
(i) 1 => True , 0 => False
(ii) "vasu" => True
(iii) 35 or any other number will give true
(iv) null => False
(v) undefined => False

Opreations

(i)
```
let value = 3
let negValue = -value
console.log(negValue)
```

Output → -3

(ii)
```
let str1 = "Hello"
let str2 = "vasu"

let str3 = str1 + str2     // concactenate string
console.log(str3)
```

Special cases
- `console.log("1" + 2)`

Output → 12

- console.log (1+"2")
  Output → 12

- console.log ("1" + 2+2)
  Output → 122

- console.log (1 + 2 + "2")
  Output → 32

These all function output is based on ECMAscript rules. For more information refer ECMAscript.

various boolean Outputs

- console.log(+true);
  Output → 1

- console.log(+"");
  Output → 0

Comparison

① 2>1
② 2>=1
③ 2<1
④ 2==1
⑤ 2 !=1

The main problem arrise when we compare two different data types.

"2" >1
"02" >1

Here on above examples there will be no issue. because JS will automatically convert data type.
But main problem arrise here →

①	null > 0	False
②	null == 0	False
③	null >= 0	True

The reason is that an equality check == and comparisons >, < , >=, <= work differently. Comparisons convert null to a number, treating it as 0.
That's why (3) null >= 0 is true and (1) null > 0 is false.

Same goes for undefined

<u>strict check (===)</u>
This will also check the data type that it should also be same

<u>eg</u>→ console. log ('2' == 2)
→ True

console. log ('2' === 2)
→ False