



Learning JavaScript

PRAVEEN NAIR

What is JavaScript?

Used to program the behavior of web pages
JavaScript was invented by Brendan Eich in 1995.

JavaScript code is inserted between `<script>` and `</script>` tags.

Javascript was developed by Netscape

JavaScript vs VBScript (Microsoft)

Javascript supports all browser, vbscript supports IE

Originally Sun Microsystem and now Oracle

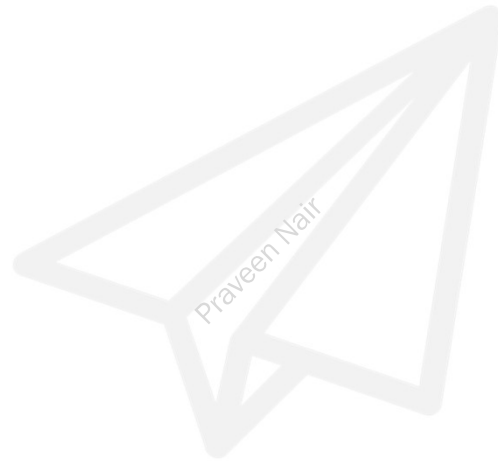
First Program

```
let a=10; // line break also works  
let b=20;  
let c = a + b;  
console.log(c);
```



Comments // and /*

```
let name='John';  
    let age=20  
    /*  
console.log(name)  
*/
```



Printing using backtick

```
let n=2;
```

```
let s = `Price of an apple is ${n}`;
```

```
document.write(s)
```

```
.....
```

Also called template literals....try multiline

Math Operators

Addition + (also concatenates string)

Subtraction -

Multiplication *

Division /

Remainder %

Exponentiation **



Comparison Operators

Operator	Description	Comparing	Returns
==	equal to	x == 8	FALSE
		x == 5	TRUE
		x == "5"	TRUE
===	equal value and equal type	x === 5	TRUE
		x === "5"	FALSE
!=	not equal	x != 8	TRUE
!==	not equal value or not equal type	x !== 5	FALSE
		x !== "5"	TRUE
		x !== 8	TRUE
>	greater than	x > 8	FALSE
<	less than	x < 8	TRUE
>=	greater than or equal to	x >= 8	FALSE
<=	less than or equal to	x <= 8	TRUE

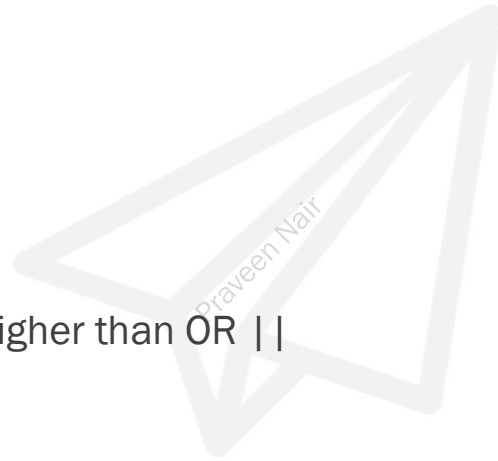
Logical Operators

Logical NOT (!)

Logical AND (&&)

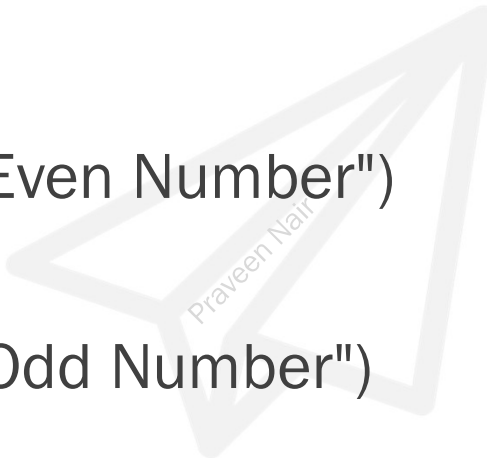
Logical OR (||)

Precedence of AND && is higher than OR ||



Conditional branching: if

```
let n = 7
if (n%2==0){
  console.log("Even Number")
}
else{
  console.log("Odd Number")
}
```



while loop

```
while (condition) {  
    ...  
}
```



For loop

```
for (let i = 0; i < 3; i++) {  
  alert(i);  
}
```

Try break and continue



JavaScript Functions

```
function showMsg() {  
    alert( 'Hello World!' );  
}
```

```
showMsg();
```



Passing arguments

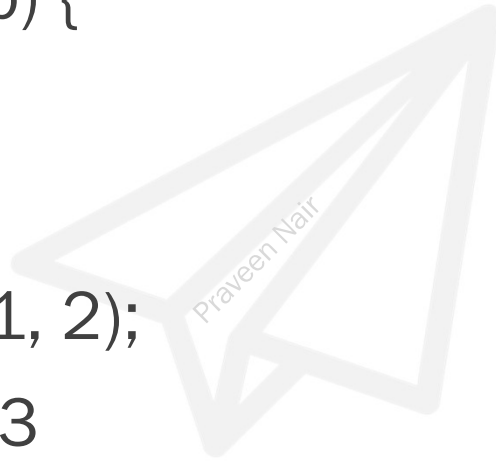
```
function sum(a, b) {  
  c = a + b;  
  alert(c);  
}  
sum(1, 2);
```



Returning Values

```
function sum(a, b) {  
    return a + b;  
}
```

```
let result = sum(1, 2);  
alert( result ); // 3
```

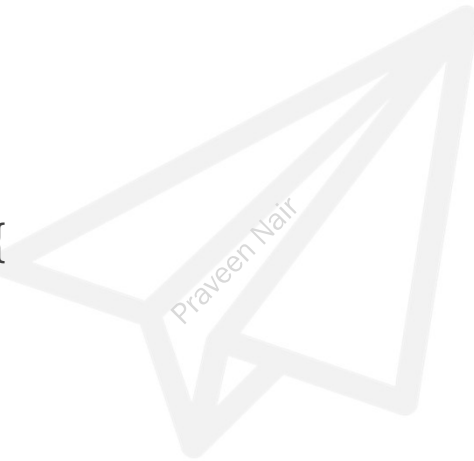


Arrow functions

```
let result = (a, b) => {  
  let c = a + b  
  return c  
};
```

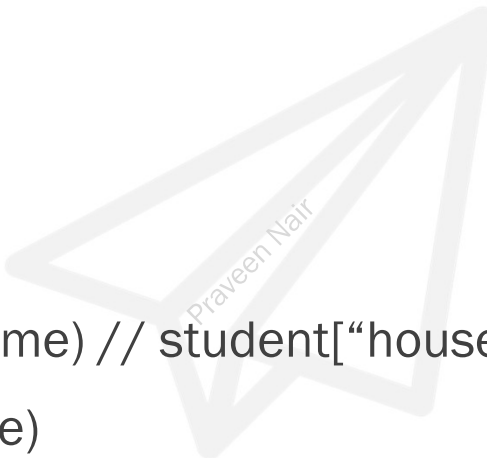
```
let result = function(a, b) {  
  let c = a + b  
  return c;  
};
```

```
alert(result(3, 2) );
```



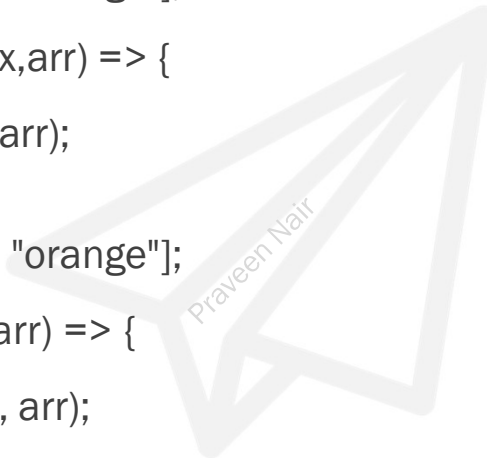
Objects – Keyed Collections

```
let student = {  
  name: "Smitha",  
  age: 30  
};  
console.log(student.name) // student["house address"]  
console.log(student.age)  
Console.log(student)
```



Array Items – foreach, map

```
let fruits = ["apple", "mango", "orange"];
  fruits.forEach((value,index,arr) => {
    console.log(value,index,arr);
  });
let fruits = ["apple", "mango", "orange"];
  fruits.map((value, index, arr) => {
    console.log(value, index, arr);
  });
```



Array Items – filter and find

```
let score = [34, 12, 67, 89, 30];  
  let result = score.filter((v) => {  
    return v > 40;  
  });  
  console.log(result);
```

```
.....  
let empnum = [1003, 1005, 1006, 1034];  
  let result = empnum.find((v) => {  
    return v == 1003;  
  });  
  console.log(result);
```

Javascript Hoisting

Hoisting is moving declarations to the top.

It is a JavaScript

.....

`x = 15;`

`elem = document.getElementById("mydiv");`

`elem.innerHTML = x;`

`var x;` 's default behavior.

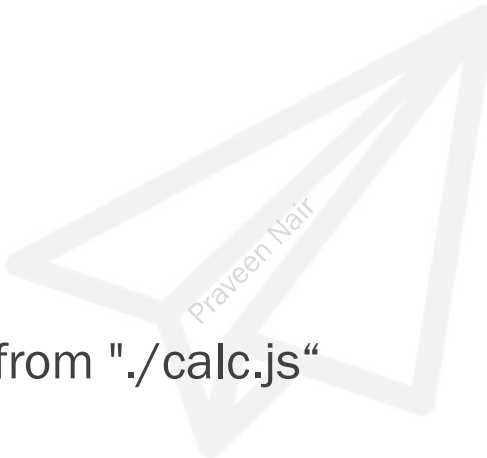
Callback functions

```
function ask(question, yes, no) {  
  if (confirm(question)) yes()  
  else no();  
}  
  
function a() {  
  alert( "You agreed." );  
}  
  
function b() {  
  alert( "You canceled." );  
}  
msg = "Do you agree?"  
ask(msg, a, b);
```



Module Import/Export - multiple

```
function add(x,y){  
  return x+y  
}  
function subtract(x,y){  
  return x-y  
}  
export {add, subtract}  
.....  
import {add,subtract} from "./calc.js"  
  let sum = add(4,5)  
  console.log(sum)  
  let difference = subtract(8,3)  
  console.log(difference)
```

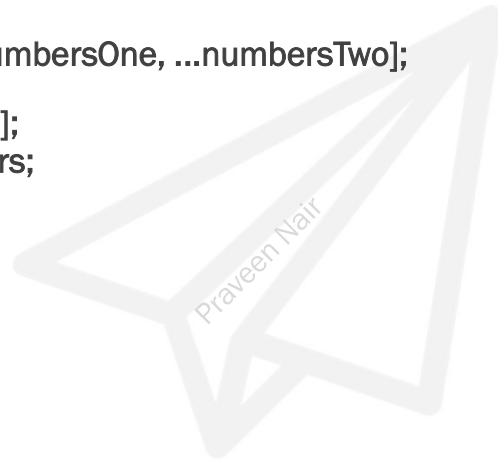


Spread Operator (...)

```
const numbersOne = [1, 2, 3];  
const numbersTwo = [4, 5, 6];  
const numbersCombined = [...numbersOne, ...numbersTwo];
```

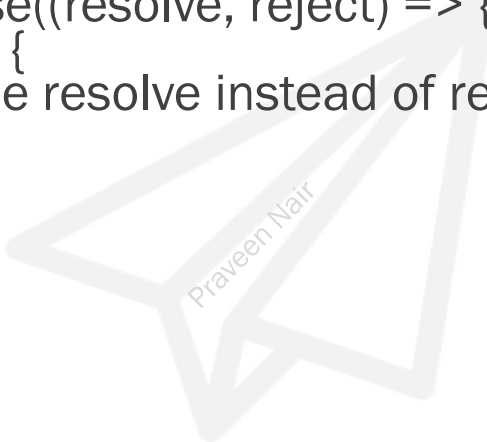
```
const numbers = [1, 2, 3, 4, 5, 6];  
const [one, two, ...rest] = numbers;
```

```
const myVehicle = {  
  brand: 'Ford',  
  model: 'Mustang',  
  color: 'red'}  
const updateMyVehicle = {  
  type: 'car',  
  year: 2021,  
  color: 'yellow'}  
const myUpdatedVehicle = {...myVehicle, ...updateMyVehicle}
```



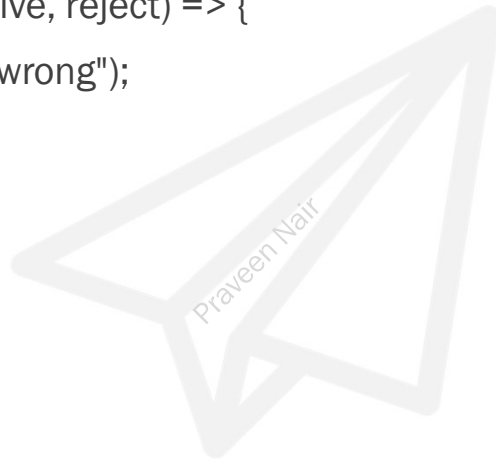
Use promise and .then

```
const f1 = () => {  
  return new Promise((resolve, reject) => {  
    setTimeout(() => {  
      resolve(5); //use resolve instead of return  
    }, 5000);  
  });  
};  
  
const f2 = (x) => {  
  console.log(x + 6)  
};  
  
f1().then((a) => f2(a));
```




Async/await

```
const f1 = () => {  
  return new Promise((resolve, reject) => {  
    // resolve(5);  
    reject("Something went wrong");  
  });  
};  
const f2 = () => {  
  console.log("Function 2");  
};  
const f3 = async () => {  
  try {  
    let n1 = await f1();  
    f2(n1);  
  } catch (err) {  
    console.log(err);  
  }  
};  
f3()
```



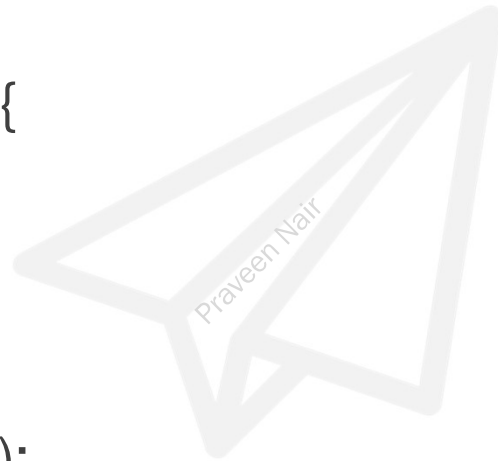
Fetch with async await

```
<body>
<ul id="userDiv"></ul>
<script>
// const url = "https://jsonplaceholder.typicode.com/users/";
const url = "students.json"
const showUsers = async () => {
  try {
    const response = await fetch(url);
    const json = await response.json();
    json.map((element) => {
      let li = document.createElement("li");
      li.innerHTML = element.name;
      userDiv.append(li);
    });
  } catch (error) {
    console.log(error);
  }
};
showUsers();
</script>
</body>
```



Closure (access to outer variable)

```
function main() {  
  let b = 1;  
  function sub() {  
    return b;  
  }  
  return sub;  
}  
let f1 = main();  
console.log(f1());  
console.log(f1());
```



Thank You

- PRAVEEN NAIR