22)Undefined and null

These are 2 new data types in js. Both these represent the absence of value.

Whenever we define a variable and we dnt assign it a value then it has value of undefined.

let name;

console.log(name);

let square = function(num) {

console.log(num);

}

square();

in function call, if we do not provide a argument to function, then undefined is passed to it. Like shown above.

If we store return value of function in a variable and that function is not returning anything, then that variable has value of undefined. code-

let square = function() {

console.log(num);

}

let result = square();

so that was undefined. Lets see null.

Lets say I have a value in a variable. Now we want to clear this value.

let age = 27;

age= undefined;

problem with this approach is, we dnt know age is undefined because we never gave it a value or we assigned it undefined explicitly. So we lost the context between the actual javascript language assigning a value and us explicitly assigning it a value. This context can be important. So to preserve this context javascript gave us access to a different type that also represents a sort of emptiness. It is null. So we can rewrite code above like this-

let age = 27;

age= null;

so when we see undefined we know it’s a language default, when we see null we know something was explicitly cleared by developer.

23)Multiple Arguments and Argument Default

let getScoreText = function(name = 'Annomous', score = 0) {

console.log(name);

console.log(score);

}

getScoreText();

getScoreText('sood', 25);

here we gave default values to our function. in first call default values will be used. In second case values that we have provided will be used.

Lets say we want to use first arguments’ default value and we want to provide value to second argument. Then we call function like this-

getScoreText(undefined, 25);

24)Function Scope

We have learned that one way to create local scope is via if statement. Now functions also define new scope. All variable that we create inside function are in that scope. This also includes arguments of function. so arguments of functions are also bound to scope of function even though they are not defined inside of curly braces.

25)template Strings

Here we will see new way to create strings. instead of combining string like this –

let name = 'sumeet';

let score = 100;

let s1 = 'Name' + name + '-Score: ' + score;

we can use template strings-

let name = 'sumeet';

let score = 100;

// let s1 = 'Name' + name + '-Score: ' + score;

let s1 = `Name: ${name} - Score ${score}`;

we instead of variables we can also do some calculations inside ${} like-

${number \*10}