**Module 22 Notes**

Video 1 –

1. ES6 or ES2015 released in 2015 is an upgraded version of JavaScript. ES mean EcmaScript.
2. Created a repository to practice ES6 and linked the local folder.

Video 2 –

1. In ES6, when declaring a variable we will be using either const or let instead of var. We used var in old JavaScript but in ES6 we will use const or let.
2. When I declare a variable which has a value that will never change then I should use const to declare it. Const is the short form of constant.
3. If I declare an array with few numbers as value, then I can replace a value, I can delete a value or I can add another value but what I can’t do is I can’t change the entire array.
4. When I declare a variable which has a value that may change then I should use let instead of var.
5. When I declare a variable with let than it’ll stay within its scope and won’t work outside it. For example when we make a for loop we write –

for(let i = 0; i < 10; i++){

}

Here we should write let instead of var. Then this let variable i will not work or show outside the for loop. If we try to display it outside the for loop with a console.log() then it’ll show an error saying i is not defined. So here the scope of i variable is for loop as we have used let to declare it.

1. So this is an important difference between var and let. Var can work outside as it doesn’t have any scope and let won’t work outside its scope. Let is leak proof.

Video 3 –

1. Suppose I am given two parameters and my task is to add these numbers. But if I mistakenly give 1 value instead of 2 then it’ll show an error. I can solve this in many ways –

function add(num1, num2){

if(num2 == undefined){

num2 = 0;

}

return num1 + num2;

}

const total = add(15);

console.log(total);

This way I can make condition that if there is no value 2 then the default value will be 0.

function add(num1, num2){

num2 = num2 || 0;

return num1 + num2;

}

const total = add(15);

console.log(total);

Then with this I can do same things which is num2 can be an actual value and if it’s not then it will be 0.

function add(num1, num2 = 0){

return num1 + num2;

}

const total = add(15);

console.log(total);

So these are 3 ways to set a backup parameter or default parameter so that the code doesn’t break for any special case.

Video 4 –

1. I can give white space between words in some ways –

const firstName = "Justin";

const lastName = "TimberLake";

const fullName = firstName + " " + lastName + " " + "is a famous person";

console.log(fullName);

const firstName2 = "Justin";

const lastName2 = "TimberLake";

const fullName2 = `${firstName2} ${lastName2} is a famous person`;

console.log(fullName2);

Output –

Justin TimberLake is a famous person

Justin TimberLake is a famous person

The second method is a ES6 feature. This is called template.

1. Now if I want to show multiple lines of sentence, the old way was–

const multiLine = "This is line number 1\n"

+ "This is line number 2\n"

+ "This is line number 3"

console.log(multiLine);

Ouput –

This is line number 1

This is line number 2

This is line number 3

But now with ES6 we can do it easily –

const multiLine2 = `This is line number 1

This is line number 2

This is line number 3`

console.log(multiLine2);

Output –

This is line number 1

This is line number 2

This is line number 3

1. So with these template feature we can do things easily. ` this symbol is called backtick which is used in ES6 template.

**Video 5 –**

1. **Function declaration –**

function doubleIt(num){

return num \* 2;

}

const result = doubleIt(5);

console.log(result);

Output –

10

**Function expression –**

const doubleIt = function(num){

return num \* 2;

}

const result = doubleIt(5);

console.log(result);

**Output –**

10

**I can also give the function a name.**

1. **I can also create a function like this in just one line. This is a shortcut. It is called arrow function –**

const doubleIt = num => num \* 2;

const result = doubleIt(5);

console.log(result);

**Output –**

10

**Here “doubleIt” is the function name. “num” is the parameter and “num \* 2” is the return value.**

**If I have multiple parameter then I must give a bracket –**

const add = (x, y) => x + y;

const result = add(50, 70);

console.log(result);

Output –

120

1. **This is an arrow function without any parameter –**

const give5 = () => 5;

const result = give5();

console.log(result);

Output –

5

1. **When I want to write a big function with arrow function then I can use {} brackets and then write the codes there –**

const doMath = (x, y) => {

const sum = x + y;

const diff = x - y;

const result = sum \* diff;

return result;

}

const result6 = doMath(7, 5);

console.log(result6);

**Output –**

24

Video 6 –

1. **I can add one array with another with .concat() method –**

const ages = [12, 14, 16, 13, 17];

const ages2 = [15, 16, 12];

const ages3 = [25, 26, 22, 27];

const allAges = ages.concat(ages2).concat(ages3);

console.log(allAges);

Output –

[

12, 14, 16, 13, 17,

15, 16, 12, 25, 26,

22, 27

]

**I can also directly add another array without declaring it –**

const ages = [12, 14, 16, 13, 17];

const ages2 = [15, 16, 12];

const ages3 = [25, 26, 22, 27];

const allAges = ages.concat(ages2).concat([15, 19]).concat(ages3);

console.log(allAges);

Output –

[

12, 14, 16, 13, 17, 15,

16, 12, 15, 19, 25, 26,

22, 27

]

**There is a much more easy way to do this –**

const ages = [12, 14, 16, 13, 17];

const ages2 = [15, 16, 12];

const ages3 = [25, 26, 22, 27];

const allAges2 = [...ages, ...ages2, 15, 19, ...ages3];

console.log(allAges2);

Output –

[

12, 14, 16, 13, 17, 15,

16, 12, 15, 19, 25, 26,

22, 27

]

**This “…” 3 dots are called spread operator or spread syntax.**

1. **We can write a function to find the maximum number –**

const business = 650;

const minister = 450;

const sochib = 250;

const maximum = Math.max(business, minister, sochib);

console.log(maximum);

Output –

650

But if we put these numbers in an array and try to find the maximum number then it’ll show an error –

const takaPoisa = [650, 450, 250];

const maximum1 = Math.max(takaPoisa);

console.log(maximum1);

Output –

NaN

It won’t work because array doesn’t work like this that anyone or in this case Math.max will read or use its value.

**But here we can use “…” to show the output correctly –**

const takaPoisa = [650, 450, 250];

// const maximum = Math.max(business, minister, sochib);

const maximum1 = Math.max(...takaPoisa);

console.log(maximum1);

Output –

650

Video 7 –

1. This way I can create a class –

class Student{

}

const student1 = new Student();

const student2 = new Student();

console.log(student1, student2);

Output –

Student {} Student {}

**Class is used to create similar type objects easily.**

1. **This is a complete example of class –**

class Student{

constructor(sId, sName){

this.id = sId;

this.name = sName;

this.school = "Random School";

}

}

const student1 = new Student(12, "Shuvo");

const student2 = new Student(22, "Mahi");

console.log(student1, student2);

Output –

Student { id: 12, name: 'Shuvo', school: 'Random School' } Student { id: 22, name: 'Mahi', school: 'Random School' }

Few things to note here –

* I must write all property inside the constructor().
* When creating a new property like id I must write “this.”
* Now for the school name I wrote a default school name. So like this if I also want to add school names then I can declare another parameter for constructor and then give the school name with id and name.

1. **I can also access any specific property from the class as it creates an object –**

Console.log(student1.name, student2.name)

Output –

Shuvo Mahi

And like this we can change a property’s value as like using an object.

1. Main benefit of class is, we can create many objects with the same structure or with same properties, and we don’t need to write those properties every time for a new object.
2. So when we need to write some repeating info for each of the element, repeating means like id or name, we can first create a class and then pass all the repeating properties (which value will change) inside constructor with “this.” And then create an object with all the informations.

Video 8 –

1. **I can inherit information from one class to other –**

class Parent {

constructor(){

this.fatherName = "Schwarzenegger";

}

}

class Child extends Parent{

constructor(name){

super();

this.name = name;

}

}

const baby = new Child("Arnold");

const baby2 = new Child("Tom");

console.log(baby);

console.log(baby2);

Output –

Child { fatherName: 'Schwarzenegger', name: 'Arnold' }

Child { fatherName: 'Schwarzenegger', name: 'Tom' }

This is called inheritance. Meaning I am getting the information from the above class to the child class by making a relation between them. The relation consists of “extends” and then the name of the class to be inherited from and then writing “super();” inside the constructor of the class which will inherit the other class.

1. I should search and learn about “object oriented programming core concept”. This consists of 3 basic things –
2. Inheritance
3. Encapsulation
4. Polymorphism
5. I can also create a function inside a class but here I don’t need to write the keyword “function” and then call the function when needed –

class Parent {

constructor(){

this.fatherName = "Schwarzenegger";

}

}

class Child extends Parent{

constructor(name){

super();

this.name = name;

}

getFullName(){

return this.name + " " + this.fatherName;

}

}

const baby = new Child("Arnold");

const baby2 = new Child("Tom");

console.log(baby.getFullName());

console.log(baby2.getFullName());

1. Function inside a class is generally called method.

Video 9 –

1. I can create an object to store various information of an element and can access specific data of the element -

const person = {name: 'Abdur Rahman', age: 20, job: 'Student', college: 'Rajshahi College', address: 'Raipara', phone: '01831531971', friends: ['Tom Hanks', 'Leonardo DiCaprio', 'Robert Downey Junior']}

console.log(person.college);

console.log(person.college);

console.log(person.college);

console.log(person.college);

Output –

Rajshahi College

Rajshahi College

Rajshahi College

Rajshahi College

**Or I can also do this –**

const person = {name: 'Abdur Rahman', age: 20, job: 'Student', college: 'Rajshahi College', address: 'Raipara', phone: '01831531971', friends: ['Tom Hanks', 'Leonardo DiCaprio', 'Robert Downey Junior']}

const collegeName = person.college;

console.log(collegeName);

console.log(collegeName);

console.log(collegeName);

console.log(collegeName);

Output –

Rajshahi College

Rajshahi College

Rajshahi College

Rajshahi College

**Another way to access a specific element –**

const person = {name: 'Abdur Rahman', age: 20, job: 'Student', college: 'Rajshahi College', address: 'Raipara', phone: '01831531971', friends: ['Tom Hanks', 'Leonardo DiCaprio', 'Robert Downey Junior']}

const { phone, college, address } = person;

console.log(phone, college, address);

Output –

01831531971 Rajshahi College Raipara

1. **I can also create an array to store similar types of data –**

const friends = ['Tom Hanks', 'Leonardo DiCaprio', 'Robert Downey Junior'];

const [firstFriend] = friends;

console.log(firstFriend);

Output –

Tom Hanks

**Access multiple data at the same time –**

const friends = ['Tom Hanks', 'Leonardo DiCaprio', 'Robert Downey Junior'];

const [firstFriend, nextFriend] = friends;

console.log(firstFriend, nextFriend);

Output –

Tom Hanks Leonardo DiCaprio

**Access rest of the remaining data –**

const friends = ['Tom Hanks', 'Leonardo DiCaprio', 'Robert Downey Junior'];

const [firstFriend, nextFriend, ...restFriend] = friends;

console.log(firstFriend, nextFriend, ...restFriend);

Output –

Tom Hanks Leonardo DiCaprio Robert Downey Junior

This will show rest of the data without the first two.

1. Destructure means the method to keep few of the data from an array or object in a variable in the shortcut way.
2. **Destructure in a complex object –**

const person = {name: 'Abdur Rahman', age: 20, job: 'Student', college: 'Rajshahi College', address: 'Raipara', phone: '01831531971', friends: ['Tom Hanks', 'Leonardo DiCaprio', 'Robert Downey Junior']}

const {phone} = person;

const complexObject = {

name: 'abc',

info: {

address: 'Somewhere',

leader: 'Some Leader'

}

}

const {leader} = complexObject.info;

console.log(leader);

Output –

Some Leader

Video 10 –

1. Things I’ve learned in this module –

* Replacing var with const and let.
* Default parameter – to give a default value for a parameter.
* Template – “ ` $ { } ` ” / “ ` ` “
* Arrow function – const functionName = (parameter) => value;

const functionName = (parameter) => {

const nameofConst = calculation or value;

const nameofConst = calculation or value;

const nameofConst = calculation or value;

return nameofConst;

{

* Arrow function is heavily used in modern javascript.
* Three dots.
* Class
* Inheritance – the way to get the element or method from the parent class to the child class.