Day 6

Date: / /

Page No.:

# Objects and functions — RR

Create object.js

- Object is a collection of properties and functionalities
- Anything in this world is an object and anything that exist in this world is an object.

Or

- collection of properties (feature that it has) & methods (work that it does)

In programming we will have
properties - variables
methods - functions.

## Functions

Create function.js

It is nothing but a block of code to perform some task.

Example:

```
let num1 = 20;
let num2 = 40;
let result = num1 + num2;
console.log(result);
```

Task: Addition

Date __/__/____ Page No.:____

If you want to use this code for multiple times so instead of repeating same thing again and again we can use function.

```
// function declaration:
function add () {
{
  let num1 = 20;
  let num2 = 40;
  let result = num1 + num2.
  console.log (result);
  }
```

Or

```
function add (num1,num2)
  {
  let result = num1 + num2;
  console.log (result);
  }
  add (30,50);
```

Or

```
// function expression.
const add = function (num1, num2)
{ let result = num1 + num2;
   console.log (result);
  };
add (30,50);
add (50,70);
```

Date __/__/____ Page No.:____

Hoisting :- Function declaration are always hoisted to the top internally.

Difference

When we create function expression then we have to create the function first then only we can access it.

Whereas when we create function declaration then we access it before its declaration. In that what is ~~do it~~ work is ~~that~~ to push the declaration at the ~~top~~. ~~This is~~ It is hoisting it to the top.

- Return
- Default Parameters

Objects

As we do in other object oriented programming language. First we create a class and then an object. here we can directly create an object and use it.

```
let obj = {
    name : "Thor",
    age : 1500,
};
console.log(obj);
```

key

{ both data and object.

Page No.:

Date __/__/____

or console.log(obj.name);
console.log(obj.age);

or
console.log(`The name is ${obj.name} and age is ${obj.age}`)

or
console.log(obj["name"]);

gives us a feature to acess value dynamically.

---

Let avenger = {
name: "Thor",
age: 1500
};
Let dynamic key = "name";
console.log(`The name is ${avenger.name} and age is ${avenger.age}`).
console.log(avenger["name"]);
console.log(avenger[dynamic key]);

to access dynamically we use this

or
Let dynamic key = "age";
console.log(avenger[dynamic key]);

Date __/__/____ Page No.: ____

```
let avenger =
{ weapons : ["Mjolnir", "Stormbreaker",
             "Thunder"],
};
console.log(avenger.weapons[0]);
or
avenger.weapons.forEach(function(weapon)
{
  console.log(weapon);
});
```

OBJECT INSIDE AN OBJECT

```
let avenger = {
  adress: {
    planet: "abcd",
    name: "xyz",
  },
};
console.log(avenger.planet.planet);
```

Function inside an Object.

```
printweapon: function()
{ console.log(this);
},
};
```

Date __/__/____

```
{console.log(this.weapons);
},
```

## Array of objects

```
let avengers = [
    {},
    {},
    {},
]
```

Three objects inside an array.

```
let avengers = [
  {
    name: "Thor",
    age: 1500
  },
  {
    name: "Captain America",
    age: 100
  },
  {   name: "Ironman",
      age: 43.
  },
]
```

Date __/__/____ Page No.:____

```
console.log(avengers[0]);
console.log(avengers[1].name);

for (i=0; i<3; i++)
  {
  console.log(avengers[i]);
  }

for (i=0; i<3; i++)
  {
  console.log(avengers[i].name);
  }

avenger.forEach(function(x) {
console.log(avengers.name);
});
```

## More about functions

### Normal function

```
function doSomething() {
  console.log("hello");
}
doSomething();
```

// arrow function :

```
doSomething = () => {
console.log ("hello");
};
doSomething ();
```

↳ This function says that if we have single line of code then we don't need curly braces.

Passing Parameters -

→ Normal function -

```
function doSomething(name)
{
    console.log ("hello" + name);
};
doSomething ("Saurab");
```

→ Arrow function.

```
doSomething = (name) => console.log ("hello"
    + name);

doSomeny ("Saurabh");
```

↳ This function also says that if we have only single parameter then we don't need () braces too.

Ex:-

```
doSomeny = name => console.log "hello" + name);
```

Maharaza

Page No.:

Date __/__/____

# Object

Normal function.

```
demo = {
  name: "Laptop",
  printname() {
    console.log();
  },
};
demo.printname();
```

this { current object }

Arrow function

```
demo = {
  name: "Laptop",
  printName: () => {
    console.log(this);
  },
};
demo.printName();
```

this { Global object }

---

Create commands.

In javascript when we work with dom browser by default gives you a lot of Object to work with dom.

window → inbuilt object
parent of all object.

Page No.:

Date __/__/____

## Child of window –

- console
- document
- screen

we can say.
window is super class.

```
console.log("hello");
or
window.console.log("hello");
```

If we don't specify the window browser automatically specify because it is the default one.

### Some Methods

```
window.alert("Hello");
   or
alert("Hello");
```

for popping up Message

```
window.prompt("Enter message");
   or
prompt("Enter message");
```

pop up but input message

prompt function return value so, we can store and retrieve it.

Date ___/___/_____ Page No.:_____

```
let name = prompt("enter name");
console.log("name");
```

```
let answer = confirm("do you want to go to
                      google");
console.log(answer);
```

(If you will press OK it will true value
else false value)

```
let answer = confirm("do you want to go to
                      google");
~~console.log(ans~~
if (answer == false)
{
console.log("okay stay here");
}
else {
window.location = "https://www.google.
                   co.in";
}
```

(In else part we are changing the url (path where we are right now)