CodeWithHarry

HTML CSS JS C C++ JAVA PYTHON PHP

Overview Q&A Downloads Announcements

ES6 Classes and Inheritance | JavaScript Tutorial In Hindi #31

In today's tutorial, we will study about ES6 Classes and inheritance. But first, let us revise the concept of OOP in javascript. In a programming language, the objects provide us with an easy way to model data. Suppose we have an object called car. The *car* object has **properties**: values that contain data about the car, and the **methods**: functions that define actions that the car can perform. The "objects" and "actions" are the basis of **Object-Oriented Programming**. The notion behind writing the code is that it more closely reflects how we think in reality. In **Object-Oriented Programming**, everything is an object, and any actions we need to perform on data are written as methods of an object. Going back to the *car* object, even in the smallest real-world application, we are going to have multiple *car* objects. If we have to code each one, we may as well go back to pen and paper.

What are the ES6 Classes?

ES6 Classes is the common JavaScript pattern of getting the class-like inheritance hierarchies using functions and prototypes. They are effectively simple compared to prototype-based OO, offering a convenient declarative form for class patterns that encourage interoperability.
Here is an example of ES6 Classes in javaScript:

```
class SimpleDate {
  constructor(year, month, day) {
    this._year = year;
    this._month = month;
    this._day = day;
  }
  addMonth(nMonths) {
  }
  getMonth() {
    return this.month;
  }
}
```

Constructors:-

The constructor method's purpose is to initialize an instance to a valid state, and it will be called automatically, so we cannot forget to initialize our objects. The constructor function is an object blueprint. From the above example, the following is the constructor of class SimpleDate:

```
constructor(year, month, day) {
    this._year = year;
    this._month = month;
    this._day = day;
  }
```

Defining Methods:-

The common practice is to assign methods directly to the prototype instead of in the initialization. But, with classes, this syntax is simplified, and we can add the method to the class. Using the method definition introduced in ES6 JavaScript, defining a method is an even more easy and concise process.

```
class Car {
 constructor(name,year ) {
        this.name = name;
        this.year = year;
}
greet() {
        return `${this.name} says hello.`;
}}
```

We will create a new instance of Car using the new keyword and assign some values.

```
const car1 = new Car('Audi', 2018);
```

Inheritance:-

Inheritance is one of the most important concepts of object-oriented programming. The significant advantage of using ES6 classes over pre-ES6 functions is that class definitions are cleaner and easy to inherit. Without writing prototype code for an inheritance, class inheritance in JavaScript is easier and similar to other object-oriented languages like Java. Following is the way of applying the inheritance concept in javaScript:

Extending a Class:-

The most useful feature of constructor functions and classes is that they can be extended into new object blueprints based on the parent. Before es6, the new constructor functions can be created from the parent using the call() method. However, with the help of ES6 classes, the super keyword is used in place of the call to access the parent functions. We will use the extends keyword to refer to the parent class.

```
//new class from the parent
class Bike extends Car {
 constructor(name,year,speed) {
       //constructor with super
       super(name, year);
       // Adding new property
       this.speed = speed;
}}
```

Now we create a new Bike instance in the same manner.

```
const bike1 = new Bike(' Trek', 2019,200);
```

Conclusion:-

In this tutorial, we learned about the similarities and differences between JavaScript constructor functions and ES6 classes. Being familiar with classes is extremely helpful, as this makes the coding easier.

Website.html code as described/written in the video

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
</head>
<body>
    <div class="container">
        <h1 id="heading" class='yourhead rhia is'> Welcome to Code With Harry</h1>
        <div id="myfirst" class="child red good" id="first">child 1

            <ul class="this" id='myul'>
```

```
                <li class="childul" id='fui'>this</li>
                <li class="childul">is</li>
                <li class="childul">a</li>
                <li class="childul">list </li>
                <li class="childul" id='lui'>of my dreams</li>
        </div>
        <div class="child">child 2</div>
        <div class="child red">child 3</div>
        <div class="child">child 4</div>
        <form action="none.html" method="post">
            <a href="//codewithharry.com">Go to Code With Harry</a>
            <br>
            <br>
            Search this website: <input type="text" name="Hello" id="">
            <button id="btn">Submit form</button>
            <!-- <input type="button" id='btn' value="submit"> -->
        </form>
    </div>
    <br>
    <div class="no">this is a dummy div1</div>
    <div class="no">this is a dummy div2</div>
    <div class="no">this is a dummy div3</div>
</body>
<!-- <script src="js/tut12.js"></script> -->
<!-- <script src="js/tut14.js"></script> -->
<!-- <script src="js/tut15.js"></script> -->
<!-- <script src="js/tut16.js"></script> -->
<!-- <script src="js/tut17.js"></script> -->
<!-- <script src="js/tut18.js"></script> -->
<!-- <script src="js/tut20.js"></script> -->
<!-- <script src="js/tut21.js"></script> -->
<!-- <script src="js/tut23.js"></script> -->
<!-- <script src="js/tut24.js"></script> -->
<!-- <script src="js/tut25.js"></script> -->
<!-- <script src="js/tut27.js"></script> -->
<!-- <script src="js/tut28.js"></script> -->
<!-- <script src="js/tut30.js"></script> -->
<script src="js/tut31.js"></script>
```

```
</html>
```

JavaScript code as described/written in the video

Copy

```
console.log("this is Tutorial31.js");

class Employee {
    constructor(givenName, givenExperience, givenDivision) {
        this.name = givenName;
        this.experience = givenExperience;
        this.division = givenDivision;
    }

    slogan(){
        return `I am ${this.name} and this company is the best`;
    }

    joiningYear(){
        return 2020 - this.experience;
    }

    static add(a, b){
        return a + b;
    }
}

class Programmer extends Employee{
    constructor(givenName, givenExperience, givenDivision, language, github){
        super(givenName, givenExperience, givenDivision);
        this.language = language;
        this.github = github;
    }

    favoriteLanguage(){
        if (this.language == 'python'){
            return 'Python';
        }
        else{
            return 'JavaScript';
        }
    }

    static multiply(a, b){
        return a * b;
    }
}

// harry = new Employee("Harry", 56, "Division");
// console.log(harry.joiningYear());
// console.log(Employee.add(34, 5))
rohan = new Programmer("Rohan", 3, "Lays", "Go", "Rohan420")
console.log(rohan)
console.log(rohan.favoriteLanguage())
console.log(Programmer.multiply(5, 7));
```

Previous Next