Name : Mayur Purushvani

Classes :

The classes is an example of ES6 version. The ES5 does not support the classes.

In ES6, We just simply write class class_name {}.

In ES5 we have to write .prototype.method_name(), but in the classes, no need for this. We just simply add a constructor and pass the default values in it.

Let's look at the below example :

Example :

```
//ES5

var Person5 = function(name,yearOfBirth, job)
{
   this.name = name;
   this.yearOfBirth = yearOfBirth;
   this.job = job;
}
Person5.prototype.calculateAge = function () {
   var age = new Date().getFullYear - this.yearOfBirth;
   console.log(age);
}

var jhon5 = new Person5('mayur', 1999, 'web developer');



//ES6
class Person6
{
   constructor(name, yearOfBirth, job)
   {
      this.name = name;
      this.yearOfBirth = yearOfBirth;
      this.job = job;
   }
   calculateAge()
   {
      var age = new Date().getFullYear - this.yearOfBirth;
      console.log(age);
   }
   static greetings()
   {
```

```
        console.log("Hello!");
    }
}

const jhon6 = new Person6('mayur', 1999, 'web developer');

Person6.greetings(); //Access the metods
```

We can also access any static methods in the class via className.methodName();

You can see the difference between ES5 and ES6 in above example.

Classes with SubClasses :

In the classes we've seen that we use extends keyword in inheritance.

So, here is one example of sub classes that inherits the parent class and access the parent property as well.

In the ES5, We use prototype, but in ES6 no need for this, We simply use super() keyword in the sub class and whatever the property of parent class have, can access the child class as well.

Here is the example of that :

```
//ES5
var Person5 = function(name,yearOfBirth, job)
{
    this.name = name;
    this.yearOfBirth = yearOfBirth;
    this.job = job;
}
Person5.prototype.calculateAge = function () {
    var age = new Date().getFullYear() - this.yearOfBirth;
    console.log(age);
}

var Athlete5 = function (name, yearOfBirth, job, olymicGames, medals) {


    Person5.call(this, name, yearOfBirth, job);
    this.olymicGames = olymicGames;
    this.medals = medals;
}
```

```
Athlete5.prototype = Object.create(Person5.prototype);

Athlete5.prototype.wonMedal = function(){
    this.medals ++;
    console.log(this.medals);
}
var jhonAthlete5 = new Athlete5('mayur', 1999, 'swimming', 12,7);

jhonAthlete5.calculateAge();
jhonAthlete5.wonMedal();



//ES6

class Person6
{
    constructor(name, yearOfBirth, job)
    {
        this.name = name;
        this.yearOfBirth = yearOfBirth;
        this.job = job;
    }
    calculateAge()
    {
        var age = new Date().getFullYear() - this.yearOfBirth;
        console.log(age);
    }
}
class Athlet6 extends Person6
{
    constructor(name, yearOfBirth, job, olymicGames, medals){

    super(name,yearOfBirth,job);
    this.olymicGames = olymicGames;
    this.medals = medals;
    }
    wonMedal()
    {
        this.medals ++;
        console.log(this.medals);
    }
}

const jhonAthlete6 = new Athlet6('mayur', 1999, 'swimming', 12, 7);
jhonAthlete6.calculateAge();
jhonAthlete6.wonMedal();
```

In above example. We can see that the We use super() keyword to access the parent property.

And in ES5 we use the call() method to access the parent property.

The biggest difference between this two is that Code reducing in ES6.

And The simple syntax is there in ES6.