ES6 Part - 2

Classes

```
//The es5 way

 var Animal = function(name) {
       this.name = name;
   }
   Animal.prototype.says = function (sound) {
       console.log(sound);
   };

  var lion = new Animal("Lion");
  lion.says("Grrrrrrrrrrrr!!!");
```

classes

```
class AnimalES6 {
    constructor(name) {
        this.name = name;
    }

   says(sound) {
        console.log(sound);
    }
}

var lionES6 = new AnimalES6("Lion");
lionES6.says(“Grrrrrrrrrrr!!!”);
```

Inheritence

Inheritance in es6 is done with the help of extends keyword

```
class Polygon {
 constructor(height, width) { //class constructor
   this.name = 'Polygon';
   this.height = height;
   this.width = width;
 }
 sayName() { //class method
   console.log('Hi, I am a', this.name + '.');
 }
}
```

```
class Square extends Polygon {
 constructor(length=10) { // ES6 features Default Parameters
   super(length, length); //call the parent method with super
   this.name = 'Square';
 }
 get area() { //calculated attribute getter
   return this.height * this.width;
 }
}

let s = new Square(5);
s.sayName(); // => Hi, I am a Square.
console.log(s.area); // => 25
console.log(new Square().area); // => 100
```

Modules

```
//Default exports

/*--- Mammal.js ---*/
export default class Mammal {...}


/*--- Human.js ---*/
Import Mammal from ‘./Mammal’;
let mammal = new Mammal();
```

TO THE NEW DIGITAL

Multiple exports and imports

```
/*--- operations.js ---*/
export function sum(a,b) { return a+b }
export function sub(a,b) { return a-b }
export function mult(a,b) { return a*b }
```

```
/*--- Main.js ---*/
Import { sum, sub, mult } from ‘./operations.js’;    //or
Import * as operations from ‘./operations’;
```

Map

```
let map = new Map()
map.set(‘name’, ‘arun’);

map.get('name')
arun

map.has('name')
true

map.delete('name')
true

map.has('name')
false
```

Map

```
// Getting Size of map
 console.log(map.size)
```

```
// Deleting all entries in a map
 map.clear()
```

Map

```
//Different ways for defining a map
let map = new Map([
        [ 1, 'one' ],
        [ 2, 'two' ],
        [ 3, 'three' ],
    ]);


let map = new Map()
    .set(1, 'one')
    .set(2, 'two')
    .set(3, 'three');
```

Iterating on maps

```
for (let key of map.keys()) {
        console.log(key);
 }

 for (let value of map.values()) {
        console.log(value);
 }

for (let entry of map.entries()) {
        console.log(entry[0], entry[1]);
 }
```

Iterating on maps

```
for (let entry of map.entries()) {
        console.log(entry[0], entry[1]);
 }
```

```
map.forEach((value, key) => {
        console.log(key, value);
});
```

Sets

```
let set = new Set();
set.add('red')

set.has('red')
true

set.delete('red')
true

set.has('red')
false
```

Sets

```
//Getting size and clearing the data from set

 let set = new Set();
 set.add('red')
 set.add('green')

 set.size
 2

set.clear();
 set.size
 0
```

Sets

Different ways of defining a set

```
let set = new Set(['red', 'green', 'blue']);
```

```
let set = new Set().add('red').add('green').add('blue');
```

Sets

```
// Iterating

for (let x of set) {
        console.log(x);
    }


// or convert it into an array and iterate over it

let arr = [...set];
```

Setting up babel

1) `npm install -g babel-cli`
2) `npm install -g babel-preset-es2015`
3) `echo '{ "presets": ["es2015"] }' > .babelrc`
4) `babel <file name>`
5) `babel <file> --out-file <output file>`
6) `babel <file> --watch --out-file <output file>`