Date : 22 - 06 - 2020
Morning Session : 9am – 11 PM
By ~ Sundeep Charan Ramkumar Today

Topics: ES6 Day 2

Var the scope of the variable function scope , where is let and const the scope is block scope.

Everything inside curly braces { }, if-else cases and for loops are **block-scoped.**

```
function sayHello () {
  var a = 5
  if (a < 10) {
    let b = 6
    var c = 10;
  }
  console.log(a, b, c)
}

sayHello()
```

```
Uncaught ReferenceError: b is not defined        app.js:11
    at sayHello (app.js:11)
    at app.js:14
```

```
console.log(iceCream);

let iceCream;
```

```
Uncaught ReferenceError: Cannot access        app.js:4
'iceCream' before initialization
    at app.js:4
```

1) **Let is a block scope variable which is under a block , it can not access beyond block**

2) **Let can not access before initialization .**

Const : Constant: something that does not change

```
const API_URL = "https://reqres.in"
API_URL = 3
```

```
Uncaught TypeError: Assignment to constant variable.    app.js:9
    at app.js:9
```

Const doesn't allow redeclaration or reassignment

Destructuring: Destructuring assignment allows for instantly mapping an object or array into many variables.

```
const numbers = [4, 5, 6, 7];
// let firstNumber = numbers[0];
let [firstNumber, secondNumber, , fourthNumber] = numbers;
console.log(firstNumber, secondNumber, fourthNumber);
```

```
const data = {
    id: 1,
    name: "Goku",
    species: "Saiyan",
    planet: "Vegeta",
    currentPlanet: "earth"
  };
```

```
function formatDBZCharacter({ name, species, planet, currentPlanet }) {
  return `
  Name: ${name}
  Species: ${species}
  Planet: ${planet}
  Current Planet: ${currentPlanet}
  `;
}

console.log(formatDBZCharacter(data));
```

Template Literals:

- Template literals is a way to deal with strings .
- Template literals use `backticks` to write a string within .
- Better to use with templates has multi-line concatenation of variables and expressions.

Arrow functions: There's another very simple and concise syntax for creating functions, that's often better than Function Expressions. It's called "arrow function

Here is a function written in ES5 syntax:

```
function timesTwo(params) {  return params * 2}function timesTwo(params) {
  return params * 2
}

timesTwo(4);  // 8
```

Now, here is the same function expressed as an arrow function:

```
var timesTwo = params => params * 2

timesTwo(4);  // 8
```

- Implicity return without the curly braces
- If parameter count is 0 or greater than 1, parentheses is must.
- Or else no need.

https://medium.com/@sgg2123/arrow-functions-in-js-ff54f558185b

New String Methods:

Sr.No	Method & Description
1	String.prototype.startsWith(searchString, position = 0) Returns true if the receiver starts with searchString; the position lets you specify where the string to be checked starts.
2	String.prototype.endsWith(searchString, endPosition = searchString.length) Returns true if the receiver starts with searchString; the position lets you specify where the string to be checked starts.
3	String.prototype.includes(searchString, position = 0) Returns true if the receiver contains searchString; position lets you specify where the string to be searched starts.
4	String.prototype.repeat(count) Returns the receiver, concatenated count times.

https://www.tutorialspoint.com/es6/es6_new_string_methods.htm