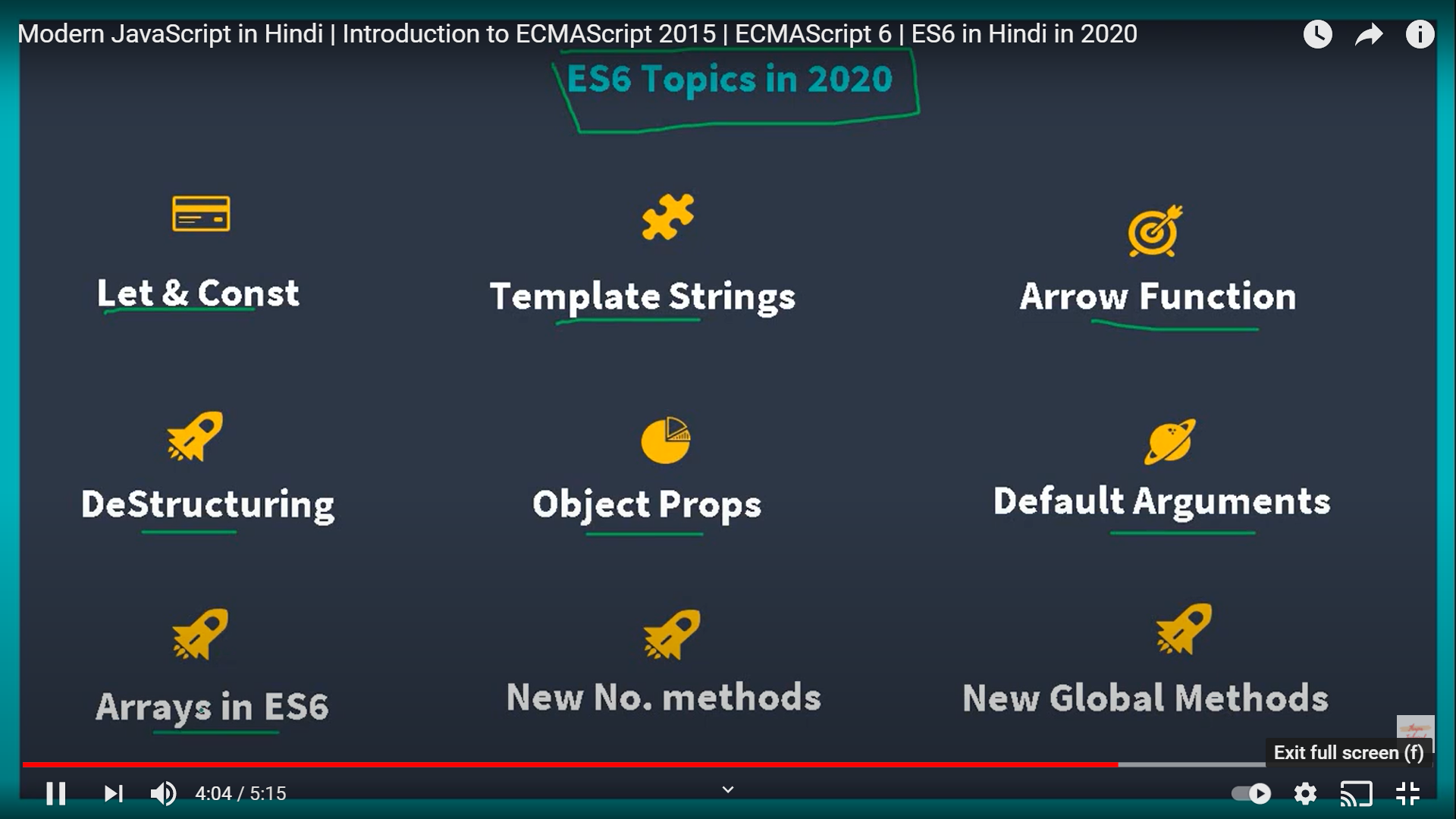
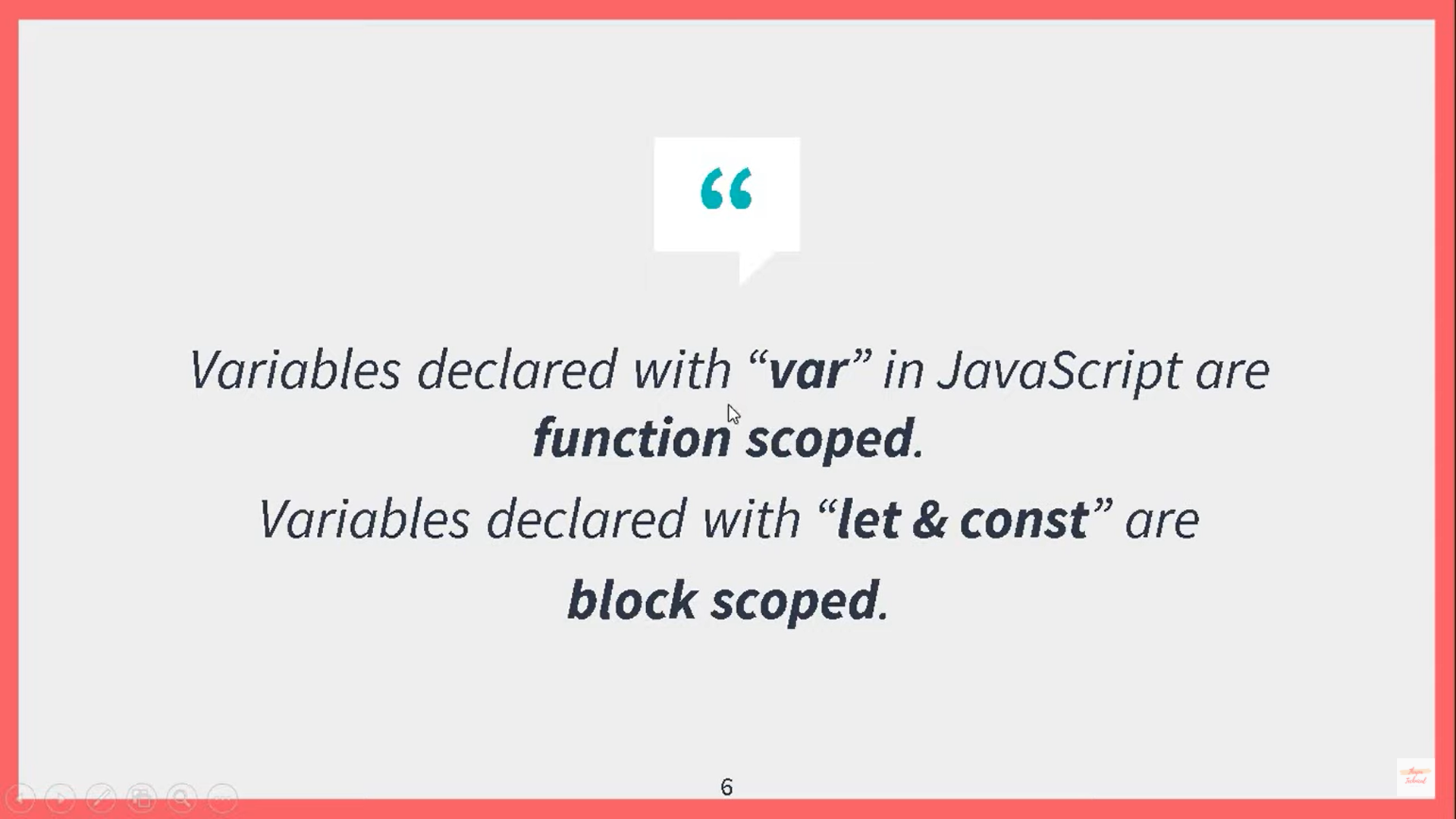


ECMAScript 6 was the second major revision to JavaScript.

ECMAScript 6 is also known as ES6 and ECMAScript 2015.





Before ES2015, JavaScript had only two types of scope: **Global Scope** and **Function Scope**.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title></title>

<script>

**var x = 20;** // It has global scope

function one() {

var x = 10;

console.log(x);

}

**console.log(x);**

</script>

</head>

<body onload="one()">

Working with Variables

</body>

</html>

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title></title>

<script>

function one() {

**var x = 10; // local scope**

console.log(x);

}

**console.log(x); // Will give error here**

</script>

</head>

<body onload="one()">

Working with Variables

</body>

</html>

Variables declared with the var keyword cannot have **Block Scope**.

Variables declared inside a block **{}** can be accessed from outside the block.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title></title>

<script>

var x = 20;

function one() {

**{**

**var x = 10;**

**console.log(x);**

**}**

**x = 100;**

console.log(x);

}

console.log(x);

</script>

</head>

<body onload="one()">

Working with Variables

</body>

</html>

Let

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title></title>

<script>

function one() {

{

let x = 10;

console.log(x);

}

x = 100;

console.log(x);

}

console.log("There is error" + x);

</script>

</head>

<body onload="one()">

Working with Variables

</body>

</html>

Const

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title></title>

<script>

function one() {

{

const x = 10;

console.log(x);

x = 1000;

console.log(x);

}

x = 100;

console.log(x);

}

console.log("There is error" + x);

</script>

</head>

<body onload="one()">

Working with Variables

</body>

</html>

Arrow Functions

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title></title>

</head>

<body>

Working with Variables

<p id="p1"></p>

**<script>**

**var res = (x, y) =>**

**x+y;**

**console.log(res);**

**document.getElementById("p1").innerHTML = res(3, 5);**

</script>

() => {}

</body>

</html>

The For/Of Loop

The For/Of Loop

The JavaScript for/of statement loops through the values of an iterable objects.

for/of lets you loop over data structures that are iterable such as Arrays, Strings, Maps, NodeLists, and more.

The for/of loop has the following syntax:

for (*variable* of *iterable*) {  
  // *code block to be executed*  
}

*variable* - For every iteration the value of the next property is assigned to the variable. *Variable* can be declared with const, let, or var.

*iterable* - An object that has iterable properties.

<!DOCTYPE html>

<html>

<body>

<h2>JavaScript For/Of Loop</h2>

<p>The for/of statement loops through the values of an iterable object.</p>

<p id="demo"></p>

<script>

let cars = ["BMW", "Volvo", "Mini"];

let text = "";

for (let x of cars) {

text += x + "<br>";

}

document.getElementById("demo").innerHTML = text;

</script>

</body>

</html>