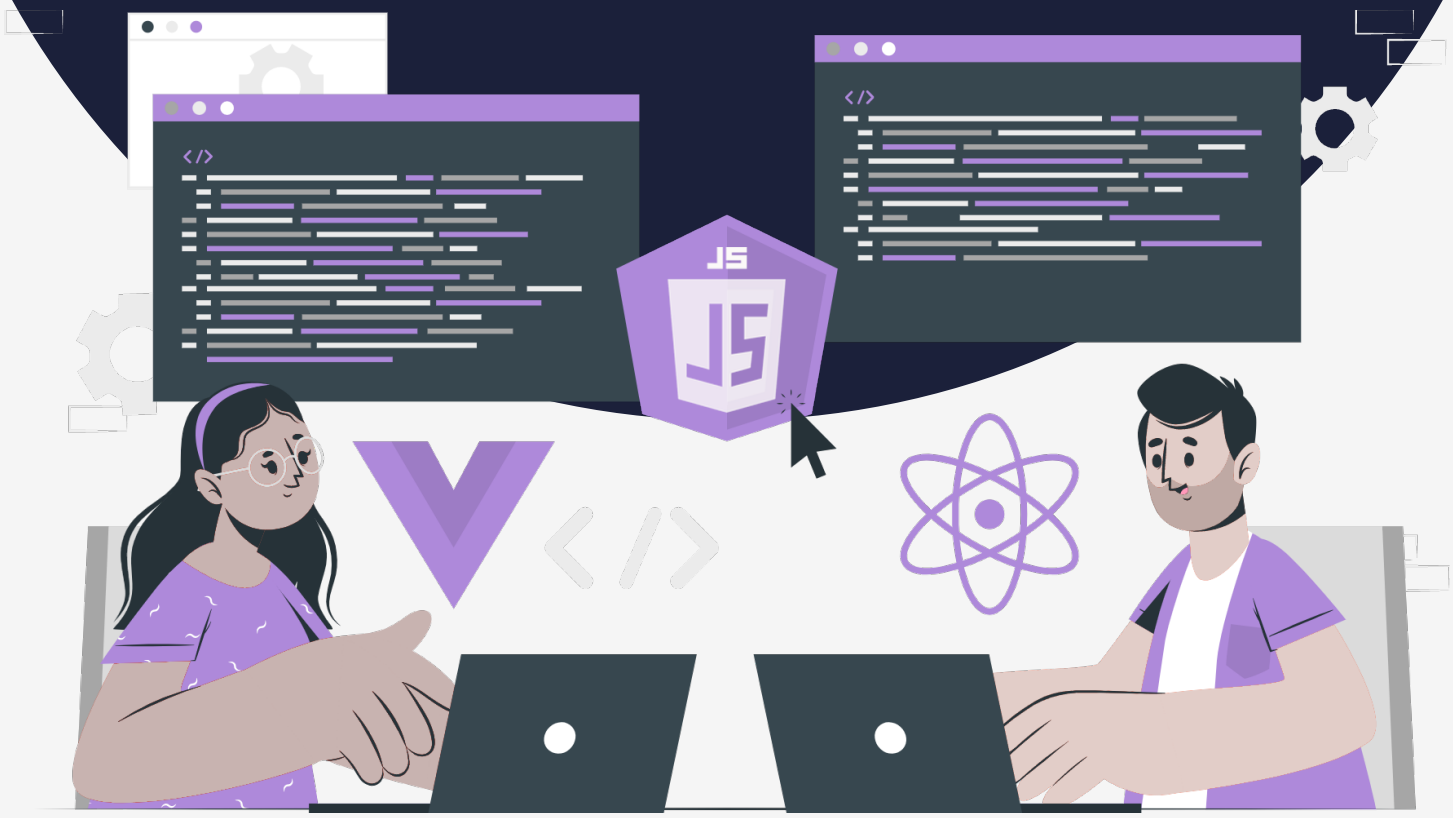
Lesson:

Destructing Arrays



Topics Covered:

1. Introduction.
2. Array Destructuring.
3. Array Destructuring for Random accessing.
4. Destructuring array with default values.
5. Swap data using Destructuring syntax.

Arrays are powerful data structures that can store values efficiently. Accessing and manipulating these values are essential to make better use of them. From the previous lectures, we have seen the introduction to the array, iterating over an array, and the array methods. In this lecture, we will be looking into the Destructuring of arrays which is a very important concept to make better use of arrays.

Destructuring means destroying or reducing to small chunks. By saying array destructuring javascript, it means to break the array into simple fragments which can be used to assign a new variable. On Destructuring the array it would be very much easier to access or reference the array items.

The Destructuring assignment makes it possible to unpack values from arrays, or properties from objects, into distinct variables. The array Destructuring uses the index in its assignment. We can extract a value from an array and put them into other variables. Array destructuring javascript is also used to assign and declare a variable.

Let's now look at the syntax, through a basic example of Destructuring an array.

```
// Array Destructuring
// Variable Declaration

let var1, var2;

// Destructuring Assignment

[var1, var2] = ["value1", "value2"];

console.log(var1); // value1
console.log(var2); // value2
```

We can also use a simpler syntax of array Destructuring too.

```
// Array Destructuring
// Destructuring Assignment

let [var1, var2] = ["value1", "value2"];

console.log(var1); // value1
console.log(var2); // value2
```

Here, in the first example we first declared two variables, var1, and var2, without initializing them. Then we used array destructuring to extract values from the array ["value1", "value2"] and assign them to var1 and var2, respectively.

In the second example, we extract values from an array and assign them to separate variables. In the example, var1 is assigned the value of "value1" and var2 is assigned the value of "value2".

Array Destructuring for Random accessing.

Let's now imagine a case where we want to access random elements from an array. We can do this by loops, but Destructuring an array would be the better option for it.

```
// Array Destructuring for Random accessing.

let techStack = ["HTML", "Express", "React", "CSS", "Javascript", "Git", "Node"]
```

Now if we want to collect the tech stack to create a static website we just need to access HTML, CSS, and Javascript which is at index 0, 3 and 4. Let's do this using array destructuring.

```
// Array Destructuring for Random accessing.

let techStack = [
  "HTML",
  "Express",
  "React",
  "CSS",
  "Javascript",
  "Git",
  "Node",
];

let [techStack0, , , techStack3, techStack4] = techStack;

console.log(techStack0); // HTML
console.log(techStack3); // CSS
console.log(techStack4); // Javascript
```

As we need only items in index 0, 3 & 4, the syntax extract specific values from the techStack array and assign them to separate variables. The syntax skips the first, second, fifth, and sixth indexes of the array.

We specify a space in the destructuring syntax to unpick elements. Whenever a space has encountered the element at that index is skipped.

Destructuring array with default values.

Sometimes, during Destructuring of an array if the value is not specified or the value is undefined the variable stored is undefined which is not expected. To avoid this we use default values.

```
let var1, var2;
[var1, var2] = [, "NewValue"];

console.log(var1); // undefined
console.log(var2); // NewValue
```

Varl stores undefined as no value is passed to it. To avoid this we use the default values.

```
// Destructuring array with default values.

let var1, var2;
[var1 = "defaultValue1", var2 = "defaultValue2"] = [, "NewValue"];

console.log(var1); // defaultValue1
console.log(var2); // NewValue
```

We can also use Destructuring syntax to swap array values.

```
// Destructuring syntax to swap array values.

let var1 = "value1", var2 = "value2";
[var1, var2] = [var2, var1];

console.log(var1); // value2
console.log(var2); // value1
```