Destructuring

What is Destructuring?

→ Destructuring is breaking down a complex structure into simpler parts. In JavaScript, the complex structure is usually an array or an object. With the destructuring syntax, we can extract smaller fragments from arrays and objects. The destructring syntax is also used for variable declaration or variable assignment.

Object Destructuring :

→ The object destructuring is a useful feature of JavaScript to extract properties from objects and bind them to variables. An object destructuring is also capable of extracting multiple properties in one statement and can access properties from nested objects. It sets a default value if the property doesn't exist. Let us take a look at what problem JavaScript destructuring solves. Sometimes, we need top-level variables like:

```
CONST PERSON = {
```

```
  FIRST: 'JOHN',
  LAST: 'ADDISON',
  COUNTRY: 'UK',
  TWITTER: '@JOHN-ADDISON'
};
CONST FIRST = PERSON.FIRST;
CONST LAST = PERSON.LAST;
```

We get this repetitive code over and over again, where we need to make a variable from something that is inside of an object or inside of an array. Here, instead of creating multiple variables we destructure them in a single line like:

```
CONST { FIRST, LAST } = PERSON;
```

This is the new destructuring syntax. The above code says: give us a variable first and last, and take it from the person object.

```
CONSOLE.LOG(FIRST); // JOHN
CONSOLE.LOG(LAST); // ADDISON
```

Similarly, if we also wanted the country, we would add the country into our code, like:

Date

```
const { first, last, country } = person;
```

→ Array Destructuring :

In array destructuring, we use an array literal on the left of an assignment expression. Each variable name on the array literal maps to the corresponding item at the same index on the destructured array.

Example:

```
var arr = ["hello", "world"];
// Destructuring assignment.
var [first, second] = arr;
console.log(first); // hello
console.log(second); // world
```

Output:

hello
world

In the above example, the left-hand side of the destructuring assignment is for defining what values are required to unpack from sourced variable. By using the rest operator (...) in array destructuring, we can put all the remaining elements of an array in a new array.

Date

Example

```
VAR COLORS = ["VIOLET", "INDIGO", "BLUE",
              "GREEN"] ; // destructuring assignment
VAR [a, b, ... c] = COLORS;
CONSOLE.LOG (a);
CONSOLE.LOG (b);
CONSOLE.LOG (c);
```

Output:

Violet

Indigo

['Blue', 'Green']