ES6 Rest Parameter

The rest parameter is introduced in ECMAScript 2015 or [ES6](https://www.javatpoint.com/es6), which improves the ability to handle parameters. The rest parameter allows us to represent an indefinite number of arguments as an array. By using the rest parameter, a function can be called with any number of arguments.

Before ES6, the **arguments** object of the function was used. The **arguments** object is not an instance of the Array type. Therefore, we can't use the **filter()** method directly.

The rest parameter is prefixed with three dots (...). Although the syntax of the rest parameter is similar to the spread operator, it is entirely opposite from the spread operator. The rest parameter has to be the last argument because it is used to collect all of the remaining elements into an array.

Syntax

1. function fun(a, b, ...theArgs) {
2. // statements
3. }

Example

1. function show(...args) {
2. let sum = 0;
3. **for** (let i of args) {
4. sum += i;
5. }
6. console.log("Sum = "+sum);
7. }
9. show(10, 20, 30);

**Output**

Sum = 60

All the arguments that we have passed in the function will map to the parameter list. As stated above, the rest parameter (...) should always be at last in the list of arguments. If we place it anywhere else, it will cause an error.

The rest parameter with the following syntax will cause an error.

1. function show(a,...rest, b) {
2. // error
3. };

Difference between Rest Parameter and arguments object

The rest **parameter** and **arguments** object are different from each other. Let's see the difference between the rest parameter and the arguments object:

* The **arguments** object is an array-like (but not array), while the rest parameters are array instances. The arguments object does not include methods such as **sort, map, forEach,** or **pop**, but these methods can be directly used in rest parameters.

Rest Parameters and Destructuring

Destructuring means to break down a complex structure into simpler parts. We can define an array as the rest parameter. The passed-in arguments will be broken down into the array. Rest parameter supports array destructuring only.

By using the rest parameter, we can put all the remaining elements of an array in a new array.

Let's see an illustration of the same.

Example

1. var colors = ["Violet", "Indigo", "Blue", "Green", "Yellow", "Orange", "Red"];
3. // destructuring assignment
4. var [a,b,...args] = colors;
5. console.log(a);
6. console.log(b);
7. console.log(args);

**Output**

Violet

Indigo

[ 'Blue', 'Green', 'Yellow', 'Orange', 'Red' ]

Rest Parameter in a dynamic function

[JavaScript](https://www.javatpoint.com/javascript-tutorial) allows us to create dynamic functions by using the **function** constructor. We can use the rest parameter within a dynamic function.

Example

1. let num = **new** Function('...args','return args');
2. console.log(num(10, 20, 30));

**Output**

[ 10, 20, 30 ]