The Spread Operator and Arrays

In this lesson you will learn how to use the spread operator with arrays.

WE'LL COVER THE FOLLOWING

- Creating a copy
- Using the spread operator to merge

Creating a copy

Three dots (`...`) define the spread operator. It can be used on an array to take every element of the array and to create a copy of each element and return the copied variables. If you spread an array of 3 elements, you will receive a copy of the ten elements positioned one after the other. In the following example, **line 1** is the original array. The following line, **line 2**, does the copy with the spread operator.

Line 4 demonstrates that they are a copy and do not point to the sane value.

```
const arrOneToBeSpread = [1, 2, 3]; // Original
const arrCopy = [...arrOneToBeSpread]; // Copy

console.log(arrCopy === arrOneToBeSpread);  // Not the same
```

Using the spread operator to merge

With a similar syntax, it is possible to merge two arrays to create a new one.

For example, if you have two arrays (**line 1** and **line 2**) and would like to merge them, you could create a new array by opening the square brackets and using the spread operator in front of the first array, followed by a comma and the spread operator followed by the second array before closing the square the spread operator followed by the second array before closing the square bracket. **Line 3** performs the merge.

```
const arrOneToBeSpread = [1, 2, 3];
const arrTwoToBeSpread = [4, 5, 6, 7];
const mergedArrCopy = [...arrOneToBeSpread, ...arrTwoToBeSpread];

console.log(mergedArrCopy);
```

Spreading to create a copy of an array is often needed in a system architecture where immutable variables are necessary. For example, if you are using React and Redux, you may want to use the spread operator to create a copy before altering any value. The spread operator is quick and safe.