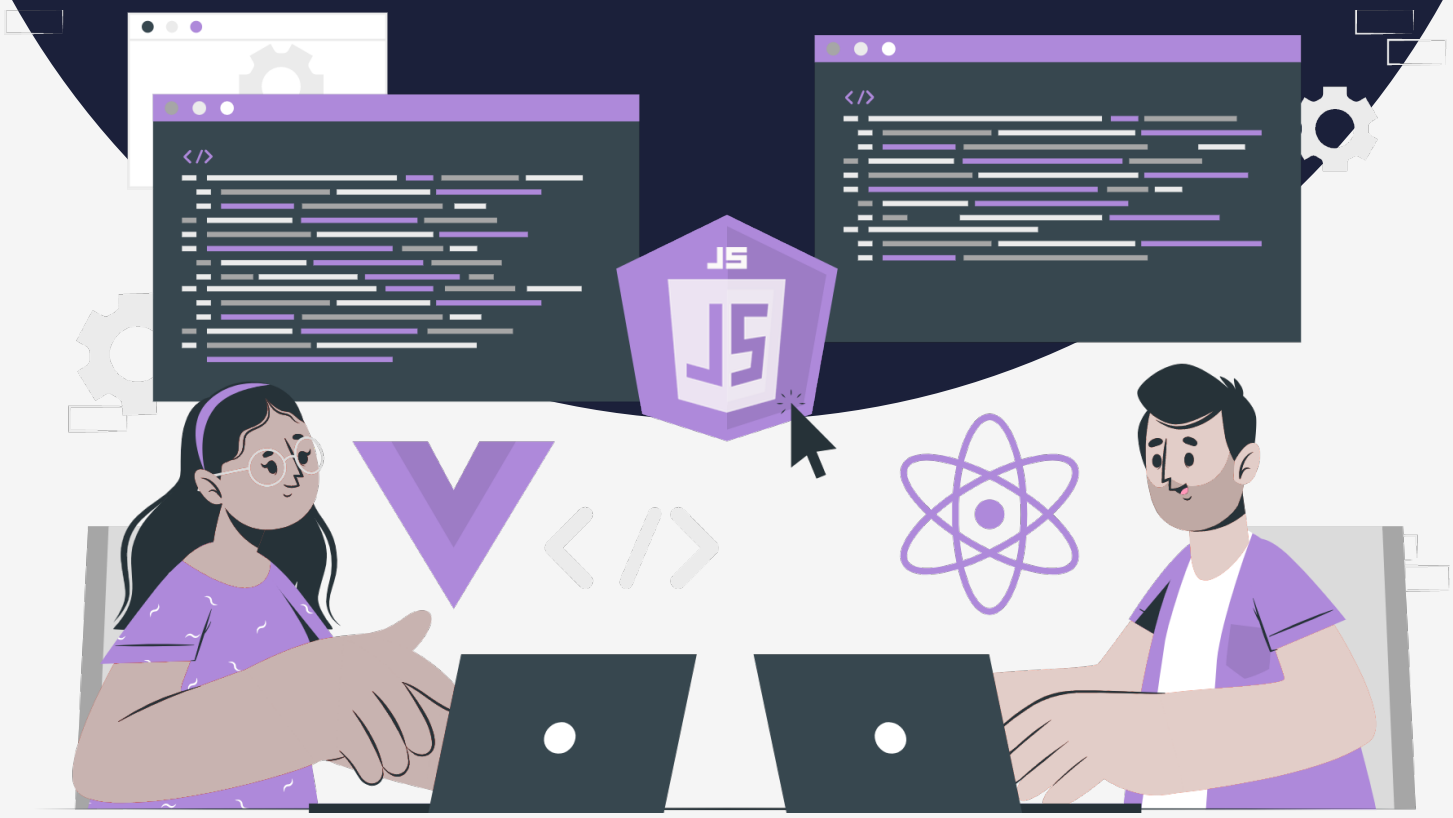
Lesson:

Spread and Rest Operator



PW SKILLS

Topics Covered:

1. Introduction.
2. Spread Operator.
3. Rest Operator.

The Rest and Spread operators in JavaScript are two powerful features that allow you to work with arrays and objects in a more flexible and concise way. The Spread operator allows you to spread elements of an array or iterable object into separate arguments, while the Rest operator gathers elements into an array. These operators provide a convenient way to manipulate and manage arrays and objects and are widely used in many programming scenarios. Let's look into these operators in this lecture.

Spread operator

The Spread operator in general allows iterables(arrays/objects/strings) to be expanded into single arguments/elements.

Let's look at some applications where spread operators can be used:

Creating a New Array

```
// Creating a New Array

let techStack = ["HTML", "CSS", "JS", "React", "Node", "Express", "MongoDB", "Git"]
let newArrayCreated = [...techStack]

console.log(newArrayCreated);

// OUTPUT: ["HTML", "CSS", "JS", "React", "Node", "Express", "MongoDB", "Git"]
```

In the above code the Spread operator allows us to spread the elements of an array into a new array, effectively creating a copy of the original array. In this code, the techStack array is spread into a new array newArrayCreated using the spread operator.

Concatenating two arrays

```
// Concatenating two arrays

let arr1 = ["HTML", "CSS", "JS"];
let arr2 = ["React", "Node", "Express"];

let concatenatedArray = [...arr1, ...arr2];
console.log(concatenatedArray);

// OUTPUT: [ 'HTML', 'CSS', 'JS', 'React', 'Node', 'Express' ]
```

In the above code the spread operator is used to concatenate two arrays by spreading the elements of both arrays into a new array. The elements of the arr1 and arr2 arrays are spread into a new array concatenatedArray using the spread operator.

Spread an array of arguments to be passed as individual params

```
function maxOfThreeNumbers(num1, num2, num3) {
  return Math.max(num1, num2, num3);
}

let arrayOfNumbers = [4, 5, 3];

console.log(maxOfThreeNumbers(...arrayOfNumbers));
// OUTPUT: 5
```

In the above code the arrayOfNumbers is passed as an argument to the sumOfThreeNumbers function to perform the sum operation.

Using with Strings

```
// Using with Strings

let name = "PW Skills";
let arrayofCharacters = [...name];

console.log(arrayofCharacters);

// OUTPUT : ['P', 'W', ' ', 'S', 'k', 'i', 'l', 'l', 's']
```

The Spread operator can be used with strings as well. In this code, a string name is declared and its characters are spread into an array arrayofCharacters using the spread operator (...name).

Spread Operator with Objects

```
// Spread Operator with Objects

let obj1 = { name: "PW Skills", course: "Full stack web development" };

let obj2 = { rating: 5, reviews: 2000 };
let newObjCreated = { ...obj1, ...obj2 };

console.log(newObjCreated);
```

```
/*
OUTPUT:
{
    name: 'PW Skills',
    course: 'Full stack web development,
    rating: 5,
    reviews: 2000
}
*/
```

The Spread operator can also be used with objects. In this code, two objects obj1 and obj2 are declared and their properties are spread into a new object newObjCreated using the spread operator (...obj1, ...obj2).

When using the spread operator on objects in JavaScript, duplicate keys in the source objects can lead to unexpected behavior.

If two or more objects being spread contain the same property key, the last property value will overwrite the previous ones.

```
// Spread Operator with Objects

let obj1 = {
  name: "PW Skills",
  course: "Full stack web development",
  numberOfStudentsEnrolled: 1000,
};
let obj2 = { rating: 5, reviews: 2000, numberOfStudentsEnrolled: 2000 };
let newObjCreated = { ...obj1, ...obj2 };
console.log(newObjCreated);

/*
OUTPUT:
{
    name: 'PW Skills',
    course: 'Full stack web development,
    numberOfStudentsEnrolled: 2000,
    rating: 5,
   reviews: 2000
}
*/
```

Adding New Values

```
// Adding New Values
let arr = ["HTML", "CSS", "JS", "React", "Node"];
let newArr = [...arr, "Git"];
console.log(newArr);

// OUTPUT: [ 'HTML', 'CSS', 'JS', 'React', 'Node', 'Git' ]

let obj = { name: "PW Skills", course: "Full Stack Web Developer" };
let newObj = { ...obj, ratings: 5 };
console.log(newObj);

// OUTPUT: { name: 'PW Skills', course: 'Full Stack Web Developer', ratings: 5 }
```

The Spread operator can be used to add new values to an array or an object. In this code, an array arr and an object obj are declared. The spread operator is used to add new values to the end of the array arr and to the object obj. The new array newArr is created by spreading the original array arr and adding the new value "Git". The new object newObj is created by spreading the original object obj and adding a new property ratings with a value of 5.

Rest Operator

The rest operator is used to collect all elements into an array. The representation is the same as a Spread operator but used differently. The rest operator gathers elements into an array.

Collecting all remaining parameters in a function.

```
// Collecting all remaining parameters in a function.

function sumOfAllNumbers(...numbers) {
  return numbers.reduce((acc, curr) => acc + curr);
}

console.log(sumOfAllNumbers(1, 2, 3, 4));

// OUTPUT: 10
```

The rest operator allows you to collect all remaining parameters in a function into an array. It's represented by three dots (...). In the example above, the rest operator (...numbers) collects all the arguments passed to the sumOfAllNumbers() function into an array called numbers. The reduce() method is then used to add up all the elements in the array and return the sum.

Destructuring

```
// Destructuring

let arr = ["HTML", "CSS", "JS", "React", "Node", "Express", "Git"];
let [element1, element2, ...remainingElements] = arr;

console.log(element1); // OUTPUT: HTML
console.log(element2); // OUTPUT: CSS
console.log(remainingElements);

// OUTPUT: [ 'JS', 'React', 'Node', 'Express', 'Git' ]

let obj = {
  name: "PW Skills",
  course: "Full Stack Web Development",
  rating: 5,
};

let { name, ...remainingProperties } = obj;

console.log(name); // OUTPUT: PW Skills
console.log(remainingProperties);

// OUTPUT: { course: 'Full Stack Web Development', rating: 5 }
```