



SPREAD OPERATOR



SPREAD OPERATOR

El operador de propagación (spread operator) permite que un **iterable** (como un array o un string) se pueda expandir en un lugar donde se esperan 0 o más **argumentos/elementos**. También permite que objetos se expanda en un lugar donde 0 o más pares **propiedad-valor** se esperan.



SPREAD OPERATOR

Pensemos en el siguiente ejemplo:

```
const sumar = (a, b) => {  
  return a + b  
};
```

```
const numerosParaSumar = [3, 5];
```

```
console.log( sumar( numerosParaSumar[0], numerosParaSumar[1] ) ); //8
```

SPREAD OPERATOR

Funciona y es relativamente simple de utilizar. Pero qué pasa si la función tiene muchos parámetros, o un número variable de parámetros.

```
const sumar = (a,b,c,d,e,f,g,h,i,j,k,ohpordio) => {  
  return a+b+c+d+e+f+g+h+i+j+k+ohpordio;  
};  
  
const numerosParaSumar = [1,2,3,4,5,6,7,8,9,10,11,12];  
  
// mmmmmmm
```

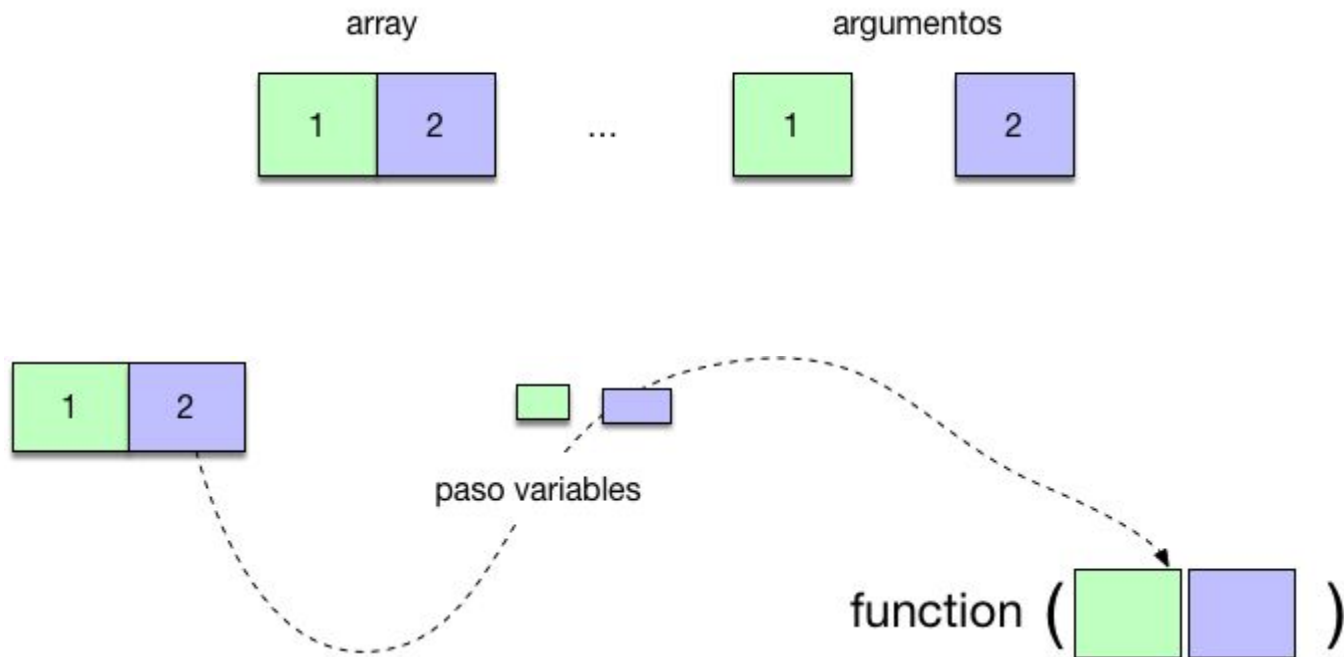
SPREAD OPERATOR

Este es un ejemplo donde puede utilizar el operador `...` (spread operator), que convierte un array en un conjunto de parámetros/argumentos.

```
const sumar = (a,b,c,d,e,f,g,h,i,j,k,ohpordio) => {  
  return a+b+c+d+e+f+g+h+i+j+k+ohpordio;  
};  
  
const numerosParaSumar = [1,2,3,4,5,6,7,8,9,10,11,12];  
  
console.log( sumar( ...numerosParaSumar ) ); // 78
```

SPREAD OPERATOR

Lo que hace el operador es **expandir** el array y convertirlo en una lista de argumentos.



SPREAD OPERATOR

También es posible utilizarlo para copiar o crear nuevos arrays.

```
const peliculas = ["Batman Begins", "The Dark Knight"];  
// NO SE COPIA EL ARRAY, SINO QUE REFERENCIA AL MISMO  
const copiaPeliculas = peliculas;  
  
copiaPeliculas.push("The Dark Knight Rises");  
  
console.log(peliculas);  
// ["Batman Begins", "The Dark Knight", "The Dark Knight Rises"]  
console.log(copiaPeliculas);  
// ["Batman Begins", "The Dark Knight", "The Dark Knight Rises"]
```


SPREAD OPERATOR

También es posible utilizarlo para copiar o crear nuevos arrays.

```
const peliculas = ["Batman Begins", "The Dark Knight"];  
// AHORA SI SE COPIA  
const copiaPeliculas = [ ...peliculas ];  
  
copiaPeliculas.push("The Dark Knight Rises");  
  
console.log(peliculas);  
// ["Batman Begins", "The Dark Knight"]  
console.log(copiaPeliculas);  
// ["Batman Begins", "The Dark Knight", "The Dark Knight Rises"]
```

SPREAD OPERATOR

Otro ejemplo donde podemos utilizar el **spread operator** para crear un array

```
// un array con dos numeros
const dosTres = [2, 3];

// creo un nuevo array, donde el primer elemento es el 1
// y le digo que expanda/propague los numeros del otro array

const unoDosTres = [1, ...dosTres];

// const unoDosTres = [1, dosTres[0], dosTres[1]];

// finalmente el array tiene los 3 elementos
console.log( unoDosTres ); // [1, 2, 3]
```

SPREAD OPERATOR: EJERCICIOS

1. Completá el siguiente código para obtener el resultado deseado en el console.log, utilizando el operador de propagación.

```
const delUnoAlTres = [1, 2, 3];  
  
const delUnoAlCinco = // ACA LA SOLUCION  
  
console.log( delUnoAlCinco );  
// [1, 2, 3, 4, 5]
```

SPREAD OPERATOR: EJERCICIOS

2. A partir del array dado, creá un nuevo array que empiece con el elemento 'CERO' y termine con el elemento 'CUATRO'

```
const arrayDado = ['UNO', 'DOS', 'TRES'];  
  
const nuevoArray = // ACA LA SOLUCION  
  
console.log( nuevoArray );  
// ['CERO', 'UNO', 'DOS', 'TRES', 'CUATRO']
```

SPREAD OPERATOR: EJERCICIOS

3. Completá el código para poder unir los 3 arrays en uno solo y llegar al resultado esperado utilizando el **spread operator**.

```
const frase0 = [';', 'Hola'];  
const frase1 = [',', ' ', 'Mundo'];  
const frase2 = ['!'];  
  
const fraseCompleta = // ACA LA SOLUCION  
  
console.log( fraseCompleta.join('') );  
// ¡Hola, Mundo!
```

SPREAD OPERATOR | OBJETOS

El **spread operator** también funciona con objetos

```
const pelicula = {  
  titulo: 'Batman Begins'  
};  
  
// NO SE COPIA EL OBJETO, SINO QUE REFERENCIA AL MISMO  
  
const copiaPelicula = pelicula;  
  
console.log( copiaPelicula.anio ); // undefined  
  
copiaPelicula.anio = 2007;  
  
console.log( copiaPelicula.anio ); // 2007  
  
console.log( pelicula.anio ); // 2007
```

SPREAD OPERATOR | OBJETOS


El **spread operator** también funciona con objetos

```
const pelicula = {  
  titulo: 'Batman Begins'  
};  
  
// ACA SI  
  
const copiaPelicula = { ...pelicula };  
  
console.log( copiaPelicula.anio ); // undefined  
  
copiaPelicula.anio = 2007;  
  
console.log( copiaPelicula.anio ); // 2007  
  
console.log( pelicula.anio ); // undefined
```

SPREAD OPERATOR | OBJETOS

También es posible crear nuevos objetos a partir de otros objetos

```
const datos = {  
  nombre: 'Ada',  
  apellido: 'Lovelace'  
};  
  
const persona = {  
  ...datos,  
  profesion: 'Programadora'  
};  
  
console.log( persona );  
  
// { nombre: "Ada", apellido: "Lovelace", profesion: "Programadora"}
```



Se copian las propiedades que estaban en el objeto **datos**

SPREAD OPERATOR: EJERCICIOS

4. Completá el siguiente código para llegar al resultado esperado (respetar orden).

```
const reviews = {  
  reviewIMDB: 9,  
  reviewFilmAffinity: 8.1  
};  
  
const peliInfo = {  
  titulo: 'The Dark Knight',  
  anio: 2008  
};  
  
const pelicula = // ACA LA SOLUCION  
console.log( pelicula );
```

```
// {  
//   titulo: "The Dark Knight",  
//   anio: 2008,  
//   reviewIMDB: 9,  
//   reviewFilmAffinity: 8.1  
// }
```

SPREAD OPERATOR: EJERCICIOS

5. Completá el siguiente código para llegar al resultado esperado (respetar orden).

```
const estudiante = {  
  nombre: 'Ada',  
  apellido: 'Lovelace'  
};  
  
const estudianteJS =// SOLUCION  
  
console.log(estudianteJS);
```

```
// {  
//   sabeJS: true,  
//   nombre: "Ada",  
//   apellido: "Lovelace",  
//   edad: 27  
// }
```

SPREAD OPERATOR: EJERCICIOS

6. La función `Math.max()` recibe varios números (distintos parámetros) y retorna el más alto. ¿Cómo modificarías el siguiente código para que funcione con distintos arrays sin tener que estar todo el tiempo usando los índices?

```
console.log( Math.max(4, 7) ); // 7
```

```
const tresNums = [9, 4, 7];
```

```
console.log( Math.max(tresNums[0], tresNums[1], tresNums[2]) ); // 9
```

```
console.log( Math.max(tresNums) ); // NaN
```

```
const masNums = [5, 5, 4, 1, 32, 132, 54, 3, 4, 5, 76, 45, 23, 65, 12, 17];
```

```
// SOLUCION
```

```
// 132
```

¡Gracias!

