

REST·SPREAD·DESTRUCT

DEVinHouse

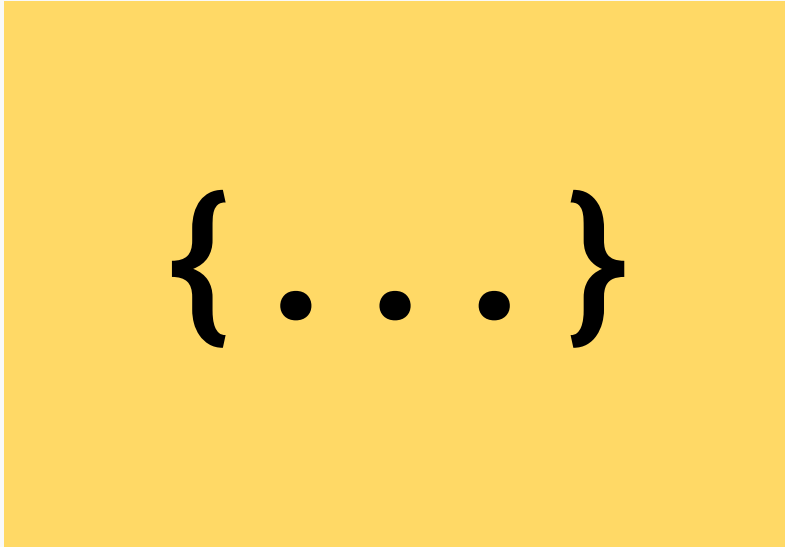
Parcerias para desenvolver a sua carreira

SENAI

<LAB365>

- **Operadores**

- Rest ...
- Spread ...
- Destructuring ...



`{ ... }`

- O operador **Rest** é usado para colocar o restante de alguns valores específicos fornecidos pelo usuário em um array, objeto ou coleção de parâmetros;
- Utiliza o símbolo três pontos (...) na sintaxe indicando que a variável nomeada agregará vários parâmetros ou itens de array/objeto

OPERADORES | Rest

```
// função comum
function infosPessoais(nome, ...infos) {
  return `${nome}: ${infos}`;
}

// arrow functions
const infosPessoais = (nome, ...infos) => {
  return `${nome}: ${infos}`;
}

const infosPessoais =
  (nome, ...infos) => `${nome}: ${infos}`;

// chamada da função
const resultado =
  infosPessoais('Maria', 'Estudante', 'Programadora');
// "Maria: Estudante, Programadora"
```

Exemplo de uso do operador rest (resto)

```
// funcao comum
function multiplicar(multip, ...numeros) {
  return numeros.map(function (n) {
    return multip * n;
  });
}

// arrow functions
const multiplicar = (multip, ...numeros) => {
  return numeros.map(n => {return multip * n});
}

const multiplicar = (multip, ...numeros) =>
  numeros.map(n => multip * n);

// chamada da função
const resultado = multiplicar(2, 4, 3, 1, 8);
// [8, 6, 2, 16]
```

Exemplo de uso com números e métodos de arrays

OPERADORES | Rest

```
// funcao comum
function encontraPares(...numeros) {
  return numeros.filter(function (n) {
    return n % 2 === 0;
  });
}

// arrow functions
const encontraPares = (...numeros) => {
  return numeros.filter(n => {return n % 2 === 0});
}

const encontraPares = (...numeros) =>
  numeros.filter(n => n % 2 === 0);

// chamada da função
const resultado = encontraPares(4, 2, 3, 1);
// [4, 2]
```

Exemplo de uso do operador rest (resto)

```
// funcao comum
function somaTodos(inicial, ...numeros) {
  return numeros.reduce(function (acum, n) {
    return acum + n;
  }, inicial);
}

// arrow functions
const somaTodos = (inicial, ...numeros) => {
  return numeros.reduce((acum, n) => {
    return acum + n;
  }, inicial);
}

const somaTodos = (inicial, ...numeros) =>
  numeros.reduce((acum, n) => acum + n, inicial);

// chamada da função
const resultado = somaTodos(4, 2, 3, 1, 8); // 18
```

Exemplo de uso com números e métodos de arrays

- O operador **Spread** "espalha" os itens de um array ou objeto, dentro de outro array/objeto ou chamada de função;
- Podemos invocar uma função que espera vários parâmetros utilizando spread;
- Spread também pode compor novos arrays e objetos;
- O operador Spread também utiliza o símbolo três pontos (...) no fragmento da sintaxe, mas dentro de arrays, objetos e chamadas de funções.

OPERADORES | Spread

```
const nomes = ['Sofia', 'Marcos', 'Bia'];

// spread no meio
const maisNomes = ['Ada', ...nomes, 'Leo'];
// ['Ada', 'Sofia', 'Marcos', 'Bia', 'Leo']

// spread no início
const maisNomes = [...nomes, 'Ada', 'Leo'];
// ['Sofia', 'Marcos', 'Bia', 'Ada', 'Leo']

// spread no fim
const maisNomes = ['Ada', 'Leo', ...nomes];
// ['Ada', 'Leo', 'Sofia', 'Marcos', 'Bia']
```

Exemplo de uso do operador spread (espalhar) com arrays

```
const infos = { nome: 'Sofia', idade: 29 };

// spread no final
const maisInfos = { nome: 'Ada', ...infos };
// { nome: 'Sofia', idade: 29 }

// spread no final
const maisInfos = { ...infos, nome: 'Ada' };
// { nome: 'Ada', idade: 29 }

// spread no início
const maisInfos = { ...infos, idade: 33 };
// { nome: 'Sofia', idade: 33 }
```

Exemplo de uso de spread com objetos

OPERADORES | Spread

```
const nomesA = ['Sofia', 'Leo'];
const nomesB = ['Marcos', 'Ada'];

// spread mantém a ordem dos itens
const todosNomes = [...nomesA, ...nomesB];
// ['Sofia', 'Leo', 'Marcos', 'Ada']

// spread mantém a ordem dos itens
const todosNomes = [...nomesB, ...nomesA];
// ['Marcos', 'Ada', 'Sofia', 'Leo']

// spread para duplicar arrays
const copiaNomes = [...nomesA];
// ['Sofia', 'Leo']
```

Exemplo de uso do operador spread (espalhar) com arrays

```
// função que recebe vários parâmetros
function somaABC(a, b, c) {return a + b + c;}
// vetor de elementos
const vetor = [4, 7, 2];
somaABC(...vetor);
// retorna 13

function apresenta(nome, idade) {
  return `${nome} tem ${idade} anos.`;
}
// em funções apenas fazemos spread de arrays
const vetor = ['Juliana', 32];
apresenta(...vetor);
// retorna "Juliana tem 32 anos."
```

Exemplo de uso de spread com objetos

INTERVALO DE AULA

DEV!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

Início: 20:20

Retorno: 20:40



DESTRUCTURING



DESTRUCTURING

- Uma maneira extra de criarmos novas variáveis, mas a partir do conteúdo de um array ou objeto já existente;
- Podemos nomear elementos de dentro de arrays, obedecendo a ordem do array;
- Podemos utilizar o mesmo nome das chaves de dentro de um objeto ou renomeá-las;
- Realizamos o destruct com chaves `{ }` ou colchetes `[]` e colocamos os nomes das variáveis dentro.

DESTRUCTURING

```
// vetor original
const vet = [1, 2, 3, 4, 5];

// destruct dos 3 primeiros itens
const [batata, blah, qqrcoisa] = vet;
// batata = 1
// blah = 2
// qqrcoisa = 3

// usando destruct e rest (sempre no final)
const [a, b, ...sobrou] = vet;
// a = 1
// b = 2
// sobrou = [3, 4, 5]
```

Exemplo de destruct com arrays, podemos usar qualquer nome

```
// objeto original
const obj =
  { nome: 'Ada', idade: 27, saldo: 99 };

// destructuring objeto com nome das chaves
const { nome, idade, saldo } = obj;
// nome = 'Ada'
// idade = 27
// saldo = 99

// destructuring apenas uma chave e resto
const { nome, ...infos } = obj;
// nome = 'Ada'
// infos = { idade: 27, saldo: 99 }
```

No destruct com objetos, usamos o nome das chaves

DESTRUCTURING

```
// vetor original
const vet = ['Ada', 27, 'Programadora'];

// destruct dos itens
const [nome, idade, profissao] = vet;
// nome = "Ada"
// idade = 27
// profissao = "Programadora"

// destruct de apenas dois itens
const [batata, qqrcoisa] = vet;
// batata = "Ada"
// qqrcoisa = 27
```

Exemplo de destruct com arrays, usando qualquer nome

```
// objeto original
const obj =
  { nome: 'Ada', idade: 27, saldo: 99 };

// renomeando itens de objetos
const { nome: batata, idade: anos } = obj;
// batata = 'Ada'
// anos = 27

// renomeando apenas um item
const { nome: fulana, idade } = obj;
// fulana = 'Ada'
// idade = 27
```

No destruct com objetos, renomeando chaves

MATERIAL COMPLEMENTAR



Aula 55 Operador Spread | <https://youtu.be/ZOP4lp1u-4>

ES6+: Operador Rest/Spread | <https://youtu.be/MxS8Aq6WleI>

Parâmetros Rest | <https://youtu.be/jA4lx6lYKag>

Juntando arrays e objetos com spread | <https://youtu.be/1Y8h-R-uymM>

Desestruturação no javascript com exemplos | <https://youtu.be/ruoHSuTKp-U>

Why Is Array/Object Destructuring So Useful And How To Use It | <https://youtu.be/Nlq3qLaHCIs>

MATERIAL COMPLEMENTAR



Parâmetros Rest | https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Functions/rest_parameters

Operadores Rest e Spread | <https://www.devmedia.com.br/javascript-operadores-rest-e-spread/41200>

Rest vs Spread Operators | <https://www.freecodecamp.org/news/javascript-rest-vs-spread-operators>

Como funcionam Rest e Spread | horadecodar.com.br/2019/03/19/como-funcionam-o-rest-e-o-spread-operator

Utilizando Destructuring | <https://www.horadecodar.com.br/2019/03/26/utilizando-destructuring-no-es6>

Optional chaining (?.) - JavaScript | developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Optional_chaining

AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

[Clique aqui](#) ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.





DEVinHouse

Parcerias para desenvolver a sua carreira

OBRIGADO!



<LAB365>