ORIGAMID

MAIS JAVASCRIPT

Rest e Spread

Parâmetros

Nem todos os parâmetros que definimos são utilizados quando uma função é executada, devido a falta de argumentos. Por isso é importante nos prepararmos para caso estes argumentos não sejam declarados.

```
function perimetroForma(lado, totalLados) {
  return lado * totalLados;
}

perimetroForma(10, 4); // 40
perimetroForma(10); // NaN
```

Parâmetro Padrão (Default) ES5

Antes do ES6 a forma de definirmos um valor padrão para um parâmetro, era através de uma expressão.

```
function perimetroForma(lado, totalLados) {
  totalLados = totalLados || 4; // se não for definido, será igual à 4
  return lado * totalLados;
}

perimetroForma(10, 3); // 30
perimetroForma(10); // 40
```

Parâmetro Padrão (Default) ES6+

Com o ES6 é possível definirmos o valor de um parâmetro direto na declaração do mesmo, caso o argumento não seja passado no momento da execução.

```
function perimetroForma(lado, totalLados = 4) {
  return lado * totalLados;
}

perimetroForma(10, 5); // 50
perimetroForma(10); // 40
```

Arguments

A palavra chave `arguments` é um objeto Array-like criado dentro da função. Esse objeto contém os valores dos argumentos.

```
function perimetroForma(lado, totalLados = 4) {
  console.log(arguments)
  return lado * totalLados;
}

perimetroForma(10);
perimetroForma(10, 4, 20);
```

Parâmetro Rest

É possível declararmos uma parâmetro utilizando `...` na frente do mesmo. Assim todos os argumentos que passarmos na ativação da função, ficarão dentro do parâmetro.

```
function anunciarGanhadores(...ganhadores) {
  ganhadores.forEach(ganhador => {
    console.log(ganhador + ' ganhou.')
  });
}

anunciarGanhadores('Pedro', 'Marta', 'Maria');
```

Único Rest

Só é possível ter um único parâmetro rest e ele deve ser o último.

```
function anunciarGanhadores(premio, ...ganhadores) {
  ganhadores.forEach(ganhador => {
    console.log(ganhador + ' ganhou um ' + premio)
  });
}

anunciarGanhadores('Carro', 'Pedro', 'Marta', 'Maria');
```

Rest vs Arguments

Apesar de parecidos o parâmetro rest e a palavra arguments possuem grandes diferenças. Sendo rest uma array real e arguments um objeto Array-like.

```
function anunciarGanhadores(premio, ...ganhadores) {
  console.log(ganhadores);
  console.log(arguments);
}

anunciarGanhadores('Carro', 'Pedro', 'Marta', 'Maria');
```

Operador Spread

Assim como o rest, o operador Spread também utiliza os `...` para ser ativado. O spread irá distribuir um item iterável, um por um.

```
const frutas = ['Banana', 'Uva', 'Morango'];
const legumes = ['Cenoura', 'Batata'];

const comidas = [...frutas, 'Pizza', ...legumes];
```

Spread Argument

O Spread pode ser muito útil para funções que recebem uma lista de argumentos ao invés de uma array.

```
const numeroMaximo = Math.max(4,5,20,10,30,2,33,5); // 33

const listaNumeros = [1,13,21,12,55,2,3,43];
const numeroMaximoSpread = Math.max(...listaNumeros);
```

Transformar em Array

É possível transformar itens iteráveis em uma array real com o spread.

```
const btns = document.querySelectorAll('button');
const btnsArray = [...btns];

const frase = 'Isso é JavaScript';
const fraseArray = [...frase];
```

Exercícios

```
// Reescreva a função utilizando
// valores iniciais de parâmetros com ES6
function createButton(background, color) {
  background = background || 'blue';
  if(color === undefined) {
    color = 'red';
  }
  const buttonElement = document.createElement('button');
  buttonElement.style.background = background;
  return buttonElement;
}

const redButton = createButton();

// Utilize o método push para inserir as frutas ao final de
comidas.
const frutas = ['Banana', 'Uva', 'Morango'];
const comidas = ['Pizza', 'Batata'];
```