

HEIG-VD / INGÉNIERIE DES MÉDIAS / PROGRAMMATION WEB

# DÉCOMPOSITION ET RESTE

```
const nums = [1, 2, 3];  
const copy = nums;  
copy.push(4);  
console.log(nums);  
// [ 1, 2, 3, 4 ]
```

**Nous avons déjà vu que les tableaux et objets sont stockés et copiés par référence.**

```
const nums = [1, 2, 3];  
const copy = nums.slice();  
copy.push(4);  
console.log(nums);  
console.log(copy);  
// [ 1, 2, 3 ]  
// [ 1, 2, 3, 4 ]
```

On peut faire une copie  
(superficielle) d'un tableau avec  
la méthode `array.slice()`

```
const nums = [1, 2, 3];  
const a = nums[0];  
const b = nums[1];  
console.log(a);  
console.log(b);  
// 1  
// 2
```

**Pour extraire un element d'un tableau, vous pouvez utiliser la syntaxe par crochet...**

```
const nums = [1, 2, 3];  
const [a, b] = nums;  
console.log(a);  
console.log(b);  
// 1  
// 2
```

**Mais il existe une syntaxe de décomposition qui nous permet de faire cela plus simplement.**

**La décomposition nous permet également d'ignorer certaines valeurs.**

```
const nums = [1, 2, 3];  
const [a, ...moreNums] = nums  
console.log(a);  
console.log(moreNums);  
// 1  
// [2, 3]
```

**Si vous souhaitez copier la totalité  
un tableau (ou ce qu'il en reste),  
vous pouvez utiliser l'opérateur de  
reste `...`**

**L'opérateur reste doit être unique  
et en dernière position.**

```
const nums = [1, 2, 3];  
const [...copy] = nums;  
copy.push(4);  
console.log(nums);  
console.log(copy);  
// [ 1, 2, 3 ]  
// [ 1, 2, 3, 4 ]
```

**Vous pouvez utiliser la décomposition pour copier la totalité d'un tableau.**

```
const n1 = [1, 2, 3];  
const n2 = [4, 5, 6];  
const combined = [...n1, ...n2];  
console.log(combined)  
// [ 1, 2, 3, 4, 5, 6 ]
```

Et l'opérateur spread **...** pour les fusionner.

On peut distinguer l'opérateur reste de l'opérateur spread par sa position vis-à-vis à l'opérateur d'assignation.



```
const flight = {  
  airline: "Swiss",  
  flightNumber: "LX41",  
  departure: "ZRH",  
  destination: "LAX",  
  passengers: ["Anna", "Bijan"]  
};  
  
const { flightNumber } = flight;  
console.log(flightNumber);  
// LX41
```

**Le même principe s'applique aux objets, sauf que c'est le nom de la propriété et non pas l'index qui fait foi.**

```
const add = (...nums) => {  
  let sum = 0;  
  for (const num of nums) {  
    sum += num;  
  }  
  return sum;  
};
```

```
const res = add(2, 3, 5, 7);  
console.log(res);  
// 17
```

**On peut utiliser l'opérateur reste comme paramètre d'une fonction pour donner à cette dernière un nombre de paramètres arbitraire.**

**On appelle cela une fonction variadique.**