La décomposition d'objet

L'affectation par décomposition

L'affectation par décomposition (`destructuring`) permet d'extraire des données d'un objet.

Elle permet d'affecter des propriétés d'un objet à des variables ce qui permet de simplifier et d'améliorer la lisibilité du code.

L'élément déstructuré se situe à droite d'une affectation (donc à droite de l'opérateur `=`).

Extraire une ou plusieurs propriétés d'un objet

Vous pouvez extraire très facilement des propriétés d'un objet pour les assigner à des variables. C'est souvent très utile pour effectuer des manipulations.

A noter que nous pouvons utiliser le raccourci syntaxique de la leçon précédente :

```
const unObjet = { toto: 1, tutu: 2};
const { toto:toto,  tutu:tutu } =  unObjet;

// Équivaut à :
const { toto,  tutu } =  unObjet;
console.log(toto); // 1
console.log (tutu); // 2
```

Nous recommandons grandement l'utilisation de l'affectation par décomposition qui permet de gagner en rapidité et en lisibilité.

Vous prendrez rapidement l'habitude de la manipuler.

Utiliser un nom différent de la propriété pour les variables

Il est également possible de définir un nom différent pour la variable auquel on assigne la valeur contenu dans la propriété de l'objet.

Pour ce faire, il suffit d'indiquer un nom différent après `:` :

```
const unObjet = { toto: 1, tutu: 2};
const { toto:a,  tutu:b } =  unObjet;
console.log(a); // 1
console.log (b); // 2
```

Utiliser des valeurs par défaut

Vous pouvez également spécifier des valeurs par défaut. Ainsi, si lors de l'affectation par décomposition la propriété vaut `undefined`, c'est-à-dire qu'on ne l'a pas défini ou qu'aucune valeur n'a été assignée, alors la valeur par défaut est utilisée :

```
const monObjet = {a: 42};
const {a = 10, b = 1} = monObjet;

console.log(a); // 42
console.log(b); // 1
```

Il est bien sûr possible d'utiliser des valeurs par défaut et d'utiliser des noms différents :

```
const monObjet = {a: 42};
const {a: nouveauNom = 10, b = 1} = monObjet;

console.log(nouveauNom); // 42
```

Déstructuration d'objets imbriqués

Vous pouvez également déstructurer des objets plus complexes qui contiennent d'autres objets :

```
 const personne = {
  nom: 'Jean',
  famille: {
    pere: {
      prenom: 'Bob',
      nom: 'Dylan',
    }
  }
};

const { famille: { pere } } = personne;
console.log(pere); // Object { nom: 'Dylan',  prenom: 'Bob' }
console.log(famille); // undefined
```

Remarquez que nous déclarons ici une nouvelle constante `pere` qui a pour valeur la valeur de la propriété `pere` de l'objet `famille` imbriqué dans l'objets `personne`.

L'opérateur `rest`

L'opérateur `rest` dont la syntaxe est `...` s'utilise notamment pour l'affectation par décomposition d'objet.

Il permet d'affecter toutes les propriétés de l'objet que nous n'avons pas explicitement affectées à des variable à un objet.

Par exemple :

```
const {a, b, ...monReste } = {a: 1, b: 2, c: 3, d: 4};
console.log(a); // 1
console.log(b); // 2
console.log(monReste); // {c: 3, d: 4}
```