4 minutes

Les opérateurs et les notions de précédence et d'associativité

Les opérateurs

Les opérateurs sont des signes ayant un effet particulier en `JavaScript`, par exemple = est l'opérateur d'assignation.

Il existe plusieurs types d'opérateur : **les opérateurs d'affectation, de comparaison, arithmétiques, binaires, logiques, pour les chaînes de caractères et les autres**.

Nous n'allons pas tous les lister mais voir les principaux maintenant et les autres au fur et à mesure de nos besoins.

Vous pouvez entendre parler **d'opérateurs unaire, binaire ou ternaire**. Cela désigne le fait que l'opérateur utilise un élément, deux éléments ou trois éléments.

Par exemple : `++maVar` est un opérateur unaire car ++ utilise un unique élément.

Alors que `1 + 2` l'opérateur + s'utilise avec deux éléments.

Il existe un seul opérateur ternaire que nous verrons dans le chapitre sur les conditions.

Les opérateurs d'affectation

Un opérateur d'affectation assigne une valeur à son élément de gauche en utilisant la valeur de l'élément de droite .

Nous en avons vu un pour le moment :

```
const a = 1;
```

Copier

= assigne 1 à l'élément de gauche `a`.

Il en existe de nombreux autres mais seuls deux sont utilisés fréquemment :

```
let a = 1;
a += 2;
a -= 3;
```

Copier

+= est l'opérateur **d'affectation après addition**. Il est en fait le raccourci pour `x = x + y`.

`-=` est l'opérateur **d'affectation après soustraction**. Il est le raccourci pour `x = x - y`.

Les opérateurs de comparaison

Ils permettent de comparer deux éléments et renvoie un booléen suivant le résultat : `true` ou `false`.

Ils permettent de voir si un élément est égal, inégal, inférieur, supérieur, supérieur ou égal, inférieur ou égal à un autre élément.

Nous les verrons en détails dans la leçon suivante. Mais voyons juste un exemple :

```
const a = 1;
const b = 2;
console.log(a === b); // false
```

Copier

Les opérateurs arithmétiques

Ils permettent d'effectuer des calculs arithmétiques sur un élément ou entre deux éléments.

Ils permettent d'additionner, soustraire, multiplier, diviser ou de calculer le modulo (reste d'une opération).

https://codesandbox.io/embed/js-c3-l5-1-9ezt5

Les opérateurs binaires

Ils permettent d'interagir directement sur les représentations binaires.

Ils sont extrêmement peu utilisés et nous ne les verrons donc pas.

Les opérateurs logiques

Ils sont utilisés sur les booléens pour aider dans les raisonnement logiques.

Ils sont au nombre de trois : **`&&`, `||` et `!`**.

Par exemple `&&` signifie et. Il permet d'effectuer le raisonnement logique si l'élément 1 et l'élément 2 sont vrais alors je retourne vrai.

```
true && true; // true
false && true; // false
false && false; // false
```

Copier

Ils sont extrêmement utilisés et nous les verrons en détails.

Les opérateurs pour les chaînes de caractères

Il existe deux opérateurs permettant de **concaténer deux chaînes de caractères**.

Concaténer signifie accoler deux chaînes de caractères pour créer une nouvelle chaîne de caractères.

```
console.log("Bonjour " + "toi !"); // Bonjour toi !
let maVar = "pré";
maVar += "fixe";
console.log(maVar); // préfixe
```

Copier

Les autres opérateurs

Nous verrons **l'opérateur ternaire (?)** lorsque nous verrons les conditions.

L'opérateur virgule (,) permet d'évaluer chacun de ses éléments de la gauche vers la droite et de renvoyer la valeur de l'élément le plus à droite.

Par exemple, pour comprendre son fonctionnement (l'exemple en lui même n'a pas d'intérêt) :

```
let x = 1;
x = (x++, x); // renvoie 2
```

Copier

Nous verrons que son principal cas d'utilisation est pour les boucles.

Il existe d'autres opérateurs spéciaux comme `typeof` que nous avons déjà vu, nous les verrons au fur et à mesure de nos besoins.

La précédence et l'associativité des opérateurs

La précédence et l'associativité des opérateurs est très importante car ce sont les règles qui permettent de comprendre l'ordre d'évaluation des opérateurs en `JavaScript`.

Ces règles sont particulièrement utiles pour les conditions, pour comprendre l'évaluation d'expressions logiques complexes :

```
true || false && true || false && false // true;
```

Copier

Vous pouvez retrouver l'ensemble des règles ici.

Cela ne sert à rien de les apprendre par coeur, nous verrons au cours de la formation les règles importantes à connaître pour l'évaluation des conditions notamment.

Retenez juste pour le moment que les règles de précédence pour les opérateurs arithmétiques sont les mêmes que celles que vous avez apprises en maths : la multiplication avant l'addition etc.