Cours 5 : Les boucles

Dans le chapitre précédent, nous avons défini et rédigé un algorithme qui permet de tester si l'utilisateur a bien tapé le mot proposé par l'application. Mais il arrive régulièrement d'avoir envie de répéter du code.

Alors, comment demander à JavaScript de réaliser ces répétitions à notre place dans le code ? En utilisant des boucles !

Découvrez les boucles

Vous aurez parfois besoin de faire plusieurs fois la même action sur une variable ou sur une partie de votre code. Dans cette situation, vous devrez utiliser des boucles.

Une **boucle** est une structure conditionnelle qui permet de répéter un certain nombre de fois du code, jusqu'à ce qu'un test ne soit plus vrai.

Pour afficher plusieurs mots, on peut donc écrire :

```
const listeMots = ['Cachalot' 'Pétunia' 'Serviette']

console log(listeMots[0])

console log(listeMots[1])

console log(listeMots[2])
```

Mais imaginez maintenant que le tableau contienne tout un dictionnaire… Ça va faire beaucoup de *console.log* ! D'autant qu'à chaque ligne, une seule chose change : l'indice du tableau (0, 1, 2…). Je vous propose donc de rédiger une boucle qui va nous permettre de répéter du code, et de résoudre ce problème.

Rédigez une boucle

Il existe deux principaux types de boucles :

- La **boucle *for*** permet de répéter du code lorsque l'on sait d'avance combien de fois il faudra le répéter.

Par exemple, si nous voulons demander **exactement trois fois** à l'utilisateur d'entrer un mot.

- La **boucle *while*** permet de répéter du code autant de fois qu'il le faut pour qu'une condition ne soit plus vraie.

Par exemple, si nous voulons redemander un mot à l'utilisateur **jusqu'à ce que** ce mot soit

correct.

Comme toujours, je vous propose de découper notre problème en petites étapes. 😉

Utilisez l'instruction *for* pour répéter du code un certain nombre de fois

Commençons par écrire une boucle qui affiche 0, puis 1, puis 2. Dans ce cas, **nous savons à l'avance combien de tours de boucle** nous voulons faire, un *for* est donc tout à fait indiqué ici.

Pour rédiger notre boucle *for,* nous allons utiliser l'instruction *for* ("pour", en anglais). Cette instruction sera suivie d'une condition entre parenthèses dans laquelle on indiquera :

- le départ de la boucle ;
- au bout de combien de tours la boucle devra s'arrêter.

Voici comment la rédiger :

```
for  let compteur = 0; compteur < 3; compteur = compteur + 1) {

   console log(compteur)

}
```

- Nous avons d'abord notre *for*, suivi de trois instructions séparées par un point virgule ";".
- Ensuite, un bloc de code entre accolades : le *console.log*.
- Ce bloc de code sera exécuté à **chaque** tour de boucle.

Première instruction :

```
let compteur = 0
```

Ici nous définissons une nouvelle variable appelée "*compteur*", et qui contient 0.

Deuxième instruction :

```
compteur < 3
```

Ceci est la condition d'arrêt.

La boucle continuera **tant que** *compteur* est plus petit que 3.

Troisième instruction :

```
compteur = compteur + 1
```

À chaque tour de boucle, on fait évoluer la valeur de *compteur*. Ici, on dit que *compteur* vaut la valeur précédente de *compteur*, plus 1.

Au premier tour de boucle, *compteur* vaut 0, puis 1, puis 2, jusqu'à ce que *compteur* arrive à 3 (la condition d'arrêt) et là, on sort de la boucle.

Le bloc de code entre accolades aura donc été exécuté 3 fois avec *compteur* ayant la valeur 0, puis 1, puis 2.

Il est très courant en programmation d'utiliser une boucle *for*. Pour la taper plus vite, il existe donc une écriture un peu raccourcie. Par convention, on nomme souvent la variable compteur "*i*" (comme "indice"). Et au lieu d'écrire *i = i+1*, on utilise l'opérateur d'incrémentation ++ , qui augmente la valeur d'une variable de 1.

Cela donne :

```
for let  = 0  i < 3; i++) {

  console log(i)

}
```

Le résultat est exactement le même.

Grâce à la boucle *for*, nous avons donc répété du code, de manière à ne pas écrire plusieurs fois la ligne de code*console.log()*.

Je vous invite à revoir ces opérations dans la vidéo ci-dessous :

Utilisez l'instruction *while* pour répéter du code

La boucle *for* est utilisée lorsque l'on connaît en amont le nombre de tours à effectuer. Mais cela n'est pas toujours possible ! Dans ce cas, nous utiliserons la boucle ***while***, qui peut s'adapter à tous les cas.

Reprenons notre exemple de compteur avec un algorithme différent :

Tant que le compteur n'est pas égal à 3, on augmente le compteur de 1.

Pour rédiger cette boucle *while*, nous allons utiliser la structure conditionnelle ***while*** ("tant que", en français) qui sera suivie d'une condition entre parenthèses. Cette condition indique le moment où notre boucle doit s'arrêter.

Nous allons donc écrire :

```
let i = 0

while (i < 3) {

  console log(i)

  i++

}
```

Dans le code ci-dessus :

- on déclare la variable *i*, que l'on initialise à zéro, **avant** la boucle ;
- le *while* ne contient que la condition d'arrêt : **tant que** *i* est plus petit que 3 ;
- on incrémente *i* (*i* va gagner *+1* à chaque tour de boucle).

Attention ! Si vous oubliez d'augmenter la valeur de *i*, alors la condition *i < 3* sera toujours vraie, et vous aurez une boucle infinie ! Cela peut même faire planter votre navigateur !

En résumé

- Une boucle est une structure conditionnelle qui permet de répéter du code plusieurs fois.
- La boucle *for* permet de répéter du code pour un nombre défini de fois.
- La boucle *while* permet de répéter du code jusqu'à ce qu'une condition ne soit plus remplie.

Vous savez désormais répéter du code grâce aux boucles ! Suivez-moi dans le chapitre suivant pour utiliser des fonctions !