JavaScript

LES TABLEAUX

Les tableaux (ou array), sont une sorte de variable avec plusieurs valeurs. C'est-à-dire qu'au lieu de déclarer plusieurs variables :

```
var fruit1 = "Pomme";
var fruit2 = "Framboise";
var fruit3 = "Cerise";
```

On va à la place faire un tableau, qui se construit :

```
var fruits = ["Pomme", "Framboise", "Cerise", "Fraise"];
```

Le tableau va lister et numéroter les éléments dans l'ordre (son index) en commençant par zéro. On peut ensuite faire appel au numéro d'index d'un élément pour agir dessus. Par exemple :

```
fruits[0]; // correspond à Pomme.
fruits[1]; // correspond à Framboise.
fruits[fruits.length - 1]; // correspond à Fraise.
fruits.indexOf("Cerise") // correspond à 2.
```

`fruits.length` *correspond au nombre total d'élément dans la liste.*

JSON

JSON signifie JavaScript Object Notation (Notation des Objets JavaScript).
C'est une notation qui permet de donner plusieurs informations à une variable. On appelle cela alors un objet. Par exemple :

```
var monLivre = {
  titre : "Titre du Livre",
  autheur : "Nom de l'autheur",
  annee : 2021
};
```

On va ensuite pouvoir accéder aux données de cet objet :

```
var titreDuLivre = monLivre.titre ;
```

On peut aussi faire un tableau d'objet :

```
var mesLivres = [
  {
    titre: "Titre du livre 1",
    auteur: "Nom de l'auteur 1",
    annee:2020
  },
  {
    titre: "Titre du livre 2",
    auteur: "Nom de l'auteur 2",
    annee:2021
  }
];
```

LES BOUCLES
Les boucles sont des méthodes qui nous permettent de répéter des instructions plusieurs fois.

for of
Cette boucle a pour fonction de répéter les instructions, un nombre de fois égale au nombre total de valeur présent dans la variable de départ.
Avec cette méthode on va aussi pouvoir récupérer plusieurs valeurs et les individualiser. Pour cela on va alors déclarer une variable au moment de la boucle qui correspond à une valeur.

Par exemple, si on veut individualiser les valeurs d'un tableau :

```
var fruits = ["Pomme", "Framboise", "Cerise", "Fraise"] ;

for (var fruit of fruits) {
    console.log(fruit);
}
```

Ici, notre variable de départ est « fruits », elle se compose de quatre valeurs. La boucle va donc répéter les instructions pour chaque fruit, donc quatre fois.

Pour individualiser les fruits on déclare la variable « fruit » qui correspond à chaque fruit. On distingue donc l'ensemble du tableau et les éléments qui le composent.

Sur un objet, on va pouvoir accéder à chacune de ses informations, par exemple :

```
var mesLivres = [
   {
       titre : "Titre du Livre1",
       autheur : "Nom de l'autheur1",
       annee : 2020
   },
   {
       titre : "Titre du Livre2",
       autheur : "Nom de l'autheur2",
       annee : 2021
   }
 ]

 for (var livre of mesLivres) {
       console.log(livre.titre);
 }
```

Maintenant, si on veut **ajouter un événement sur un élément sélectionné**. Il faut dans un premier temps sélectionner plusieurs éléments, avec un sélecteur multiple comme « `querySelectorAll` ».

```
var mesElements = document.querySelectorAll('button');

for (var element of mesElements) {
       element.addEventListener("click", function(){
          this.style.backgroundColor = 'red' ;
     })
}
```

Ici on va appliquer un fond rouge sur l'élément qui est cliqué. Le mot clé « `this` »indique que c'est cet élément (celui qui est cliqué) qui aura les instructions.

Il existe d'autres boucles plus complexe qui sont utiles dans des cas particuliers. Vous pouvez les retrouver dans la documentation.

LES CONDITIONS

Il existe plusieurs sortes d'instructions conditionnelles. On les utilise pour effectuer des tâches dans le cas ou les conditions sont bien remplies.

if

Permet d'exécuter une instruction SI la condition est remplie.

```
if (condition){
  // Si la condition est remplie on exécute les instructions placées ici.
}
```

if else
Permet d'exécuter une instruction SI la condition est remplie SINON (dans le cas ou la condition n'est pas remplie) on exécute une autre instruction.

```
if (condition){
  //  Si la condition est remplie, ces instructions.
}else{
  //  Sinon on exécute les instructions placées ici.
}
```

else if
Permet d'ajouter des conditions supplémentaires dans le cas ou les précédentes ne sont pas remplies.

```
if(condition1){
  // Si la condition1 est remplie on exécute ces instructions.
}else if(condition2){
  /* Si la condition1 n'est pas remplie mais que la condition2 l'est,
  on exécute ces instructions. */
}else{
  /* Sinon, si aucunes des conditions ne sont remplies,
  on exécute les instructions placées ici. */
}
```

PETITE LISTE DES OPÉRATEURS
Les opérateurs comparatifs :

- **<** Inférieur à.
- **<=** Inférieur ou égal à.
- **==** Égal à.
- **>=** Supérieur ou égal à.
- **>** Supérieur à.
- **!=** Différent de.

Les opérateurs logiques :

- **&&** Permet de vérifier si deux conditions sont vraies (signifie « ET »).
- **||** Permet de vérifier si au moins une condition est vraie (signifie « OU »).
- **!** Permet de vérifier si une condition n'est pas vraie (signifie « NON »).

Exemple :

```
if (maVariable > 2 && maVariable < 6){
  /*
    Si la valeur de maVariable est supérieure à 2 ET est inférieure à 6,
    on exécute les instructions placées ici..
  */
}
```

<u>À noter</u> : Il est très fortement conseillé d'**utiliser la console** avec la fonction `console.log();` pour vérifier si on passe bien dans les boucles, les conditions, les fonctions, etc.