Workshop II

Les objets

Un objet possède une structure qui lui permet d'interagir avec d'autres objets.

```
var maVariable = 42;
```

En réalité, une variable contient un objet, ici maVariable contient un objet natif de type Number dont la valeur est 42. Il existe différents objets natifs : Object, Function, Boolean, Number, Date, String, Array, JSON et d'autres...

Il existe trois concepts distincts pour un objet : le constructeur, les propriétés et les méthodes.

Les objets : le constructeur

Le constructeur est un code qui s'exécute quand on initialise une nouvel objet, il permet d'exécuter différentes instructions.

```
// On initialise un objet natif String
var maVariable = 'Chaîne de caractères 42';
```

Ici pour un objet natif String c'est automatique.

Les objets : les propriétés

Une propriété est une variable contenue dans l'objet qui lui fourni des informations nécessaires à son fonctionnement.

```
// On initialise un objet natif String
var maVariable = 'Ceci est une chaîne de caractères violette';
// On utilise la propriété length
console.log(maVariable.length);
// => 42
```

Les objets : les méthodes

Une méthode est une fonction contenue dans l'objet qui lui permet de le modifier.

```
// On initialise un objet natif String
var maVariable = 'Chaîne de caractères 42';
// On utilise la méthode toUpperCase
console.log(maVariable.toUpperCase());
// => CHAÎNE DE CARACTÈRES 42
```

Les tableaux

Un tableau est un espace de stockage de plusieurs valeurs.

On appelle une valeur, un item. Chaque item est accessible avec un index qui commence à 0.

```
// On initialise un tableau
var maTable = [3, 42, 'Chaîne de caractères'];
console.log(maTable[1]);
// => 42
maTable = [[3, 42], 100, 'Chaîne de caractères'];
console.log(maTable[0][1]);
// => 42
```

Il existe différentes méthodes pour modifier un tableau : filter, findIndex, forEach, indexOf, map, pop, push, shift, slice, sort, unshift et d'autres...

Les tableaux : filter

La méthode filter crée et retourne un nouveau tableau contenant tous les items du tableau d'origine qui remplissent une condition.

```
var maTable = [0, 12, 36, 42, 77];
console.log(maTable.filter(item => item > 36));
// => Array [42, 77]
```

Les tableaux : findIndex

La méthode findIndex renvoie l'index du premier item du tableau si la condition de la fonction retourne true. Si la condition retourne false pour tous les items du tableau, la méthode renvoi -1.

```
var maTable = [0, 12, 36, 42, 77];
console.log(maTable.findIndex((function (item) {
    return item == 42;
})));
// => 3
console.log(maTable.findIndex((function (item) {
    return item > 100;
})));
// => -1
```

Les tableaux : forEach

La méthode forEach permet d'exécuter une fonction sur chaque item du tableau.

```
var maTable = [0, 12, 36, 42, 77];
maTable.forEach(function(item) {
    console.log(item);
});
// => 0
// => 12
// => 36
// => 42
// => 77
```

Les tableaux : indexOf

La méthode indexOf renvoie le premier index de l'item trouvé si la condition comparative === retourne true. Si la condition retourne false pour tous les items du tableau, la méthode renvoi -1.

```
var maTable = [0, 12, 36, 42, 42, 77];
console.log(maTable.indexOf(42));
// => 3
console.log(maTable.indexOf(100));
// => -1
```

Les tableaux : map

La méthode map crée et retourne un nouveau tableau contenant le résultat de la fonction exécutée sur chaque item du tableau.

```
var maTable = [0, 12, 36, 42, 77];
console.log(maTable.map(item => item * 2));
// => Array [0, 24, 72, 84, 154]

maTable = ['0', '12', '36', '42', '77'];
console.log(maTable.map(Number));
// => Array [0, 12, 36, 42, 77]
```

Les tableaux : pop

La méthode pop supprime le dernier item d'un tableau et le retourne.

```
var maTable = [0, 12, 36, 77, 42];
console.log(maTable.pop());
// => 42
console.log(maTable);
// => Array [0, 12, 36, 77]
```

Les tableaux : push

La méthode push ajoute un ou plusieurs items à la fin d'un tableau et retourne la nouvelle taille du tableau.

```
var maTable = [0, 12, 36, 77];
console.log(maTable.push(42));
// => 5
console.log(maTable);
// => Array [0, 12, 36, 77, 42]
```

Les tableaux : shift

La méthode shift supprime le premier item d'un tableau et le retourne.

```
var maTable = [42, 0, 12, 36, 77];
console.log(maTable.shift());
// => 42
console.log(maTable);
// => Array [0, 12, 36, 77]
```

Les tableaux : slice

La méthode slice renvoie une copie d'une portion du tableau, la portion est définie par un index de début et un index de fin (exclus). Le tableau original n'est pas modifié.

```
var maTable = [0, 12, 36, 42, 77];
console.log(maTable.slice(3));
// => Array [42, 77]
console.log(maTable.slice(2, 4));
// => Array [36, 42]
console.log(maTable);
// => Array [0, 12, 36, 42, 77]
```

Les tableaux : sort

La méthode sort tri les items à l'aide d'une structure comparative.

```
var maTable = [0, 132, 42, 36, 5, 77];
maTable.sort((a, b) => a - b);
console.log(maTable);
// => Array [0, 5, 36, 42, 77, 132]

maTable.sort((a, b) => b - a);
console.log(maTable);
// => Array [132, 77, 42, 36, 5, 0]
```

Les tableaux : unshift

La méthode unshift ajoute un ou plusieurs items au début d'un tableau et retourne la nouvelle taille du tableau.

```
var maTable = [0, 12, 36, 77];
console.log(maTable.unshift(42));
// => 5
console.log(maTable);
// => Array [42, 0, 12, 36, 77]
```

Les objets littéraux

Un objet littéral est grossièrement un tableau.

À la différence que chaque item est accessible avec une key.

```
// On initialise un objet littéral
var monObjet = {
    nom: 'Angamara',
    devise: 42
};
console.log(monObjet.devise);
// => 42
console.log(monObjet['devise']);
// => 42
var maVariable = 'devise'
console.log(monObjet[maVariable]);
// => 42
```

Les objets littéraux : Ajouter une key

Pour ajouter une key à un objet littéral existant il suffit de la définir.

```
// On initialise un objet littéral
var monObjet = {
    nom: 'Angamara',
    devise: 42
};
monObjet.jouet = 'Rubik's Cube';
console.log(monObjet.jouet);
// => Rubik's Cube
```

La boucle for of

La boucle for of permet de parcourir les items d'un objet, d'un tableau, d'une chaîne de caractères et d'autres...

```
for (variable of objet) {
    instruction;
}
```

La boucle for of : Les tableaux

Solution pour retourner directement l'item :

```
var maTable = ['Hello', 'world' ];
for (let item of maTable) {
    console.log(item);
}
// => Hello
// => world
```

Solution pour retourner l'item via son index :

```
var maTable = ['Hello', 'world' ];
for (let index of maTable.keys()) {
    console.log(maTable[index]);
}
// => Hello
// => world
```

La boucle for of : Les objets littéraux

Solution pour retourner l'item via sa key :

```
var monObjet = {
    nom: 'Angamara',
    devise: 42
};
for (let key of Object.keys(monObjet)) {
    console.log(monObjet[key]);
}
// => Angamara
// => 42
```

La boucle for of : Les chaînes de caractères

Solution pour retourner directement l'item :

```
var maVariable = 'Hello';
for (let item of maVariable) {
    console.log(item);
}
// => H
// => e
// => l
// => l
// => o
console.log(maVariable[0]);
// => H
```