**PREMIERE SEANCE**

1. Qu’est-ce que le JS (Définition)

**Le JS est un langage de script multiplate-forme et orienté objet. C’est un langage léger qui doit faire partie d’un environnement hôte (Un navigateur web par exemple) pour qu’il puisse être utilisé sur les objets de cet environnement.**

* Donc on aura pas forcément besoin d’installer le JS… Donc dites-vous que le JS est installé par défaut dans tous les navigateurs (Moi je vous recommande Chrome de Google)
* Il est léger c’est-à-dire qu’il s’exécute vite (Lorsqu’on fera nos exercices et projets vous allez mieux comprendre)
* Il y’a une chose qui n’est pas précisée dans la définition, c’est que le JS est asynchrone (Mais on va mieux y revenir après)
* C’est un langage front-end car c’est notre navigateur qui interprète le code (Notre navigateur) du fichier qui est stocké dans le serveur
* Lorsque vous faites du web, le JS est primordial
* En apprenant le JS, vous allez pouvoir apprendre REACT, REACT Native, Angular et Vue.js très rapidement

NB : Le JavaScript est différent du JAVA car le JAVA c’est pour faire du Backend

1. On peut écrire du JS dans les balises **script**
2. On peut écrire dans la console
3. On créer un fichier (.js) qu’on va lier à notre index.html

NB : Le fichier JS doit toujours être lu en dernier car le JS parfois doit être long à exécuté et que pendant le chargement de la page il faut que l’utilisateur puisse voir le contenu de la page même si le JS est encore en train de charger

1. On peut aussi déclarer le fichier dans le <head></head> mais dans ce cas il faudra utiliser le defer

# Introduction à la Syntaxe de JavaScript

## 1. Les Variables

Les variables sont des conteneurs permettant de stocker des informations que l’on peut réutiliser ou modifier plus tard. En JavaScript, on utilise principalement les mots-clés **let**, **const** et, plus anciennement, **var** pour déclarer des variables.  
  
- **let** : Permet de déclarer une variable qui peut être réassignée.  
  
 **let** age = 25; *// On déclare une variable "age" avec la valeur 25*  
 **age** = 26; // *On change la valeur de "age" à 26*  
  
  
- **const**: Utilisé pour déclarer une constante, c’est-à-dire une valeur qui ne changera pas une fois définie.  
  
 **const** nom = "Alice";  
 // nom = "Bob"; // *Cela génèrerait une erreur car "nom" est une constante et on ne peut pas la redéfinir(Affecter une nouvelle valeur)*  
  
  
- **var** : Avant **let** et **const**, var était le mot-clé principal pour déclarer des variables, mais il est moins utilisé aujourd'hui en raison de certaines limitations.

## 2. Les Types de Données

JavaScript propose plusieurs types de données que l’on peut utiliser selon les besoins :  
  
- **Number (Nombre)** : Pour représenter des nombres, comme 5 ou 3.14.  
- String (Chaîne de caractères) : Pour représenter du texte, entouré de guillemets simples ou doubles ("Hello" ou 'Bonjour').  
**- Boolean (Booléen)** : Représente des valeurs de type vrai/faux (true ou false).  
- **Array (Tableau)** : Un ensemble de valeurs regroupées dans une seule variable.  
  
***let*** *couleurs = ["rouge", "bleu", "vert"];*  
- **Object (Objet)** : Une structure complexe qui permet de stocker des propriétés en paires clé-valeur.  
  
 *let personne = { nom: "Alice", age: 25 };*

## 3. Les Opérateurs

Les opérateurs permettent de manipuler des valeurs dans JavaScript.  
  
- **Opérateurs arithmétiques:** `+` (addition), `-` (soustraction), `\*` (multiplication), `/` (division).  
  
 let somme = 5 + 3; // *somme vaut 8*  
  
  
- **Opérateurs de comparaison :** `===` (égal), `!==` (différent), `<`, `>`, `<=`, `>=`.  
  
 **let** estEgal = (5 === 5); *// vrai (true)*  
 let estDifferent = (5 !== 3); *// vrai (true)*  
 `

# La Concaténation en JavaScript

La concaténation en JavaScript consiste à assembler des chaînes de caractères pour former une seule chaîne. Elle est souvent utilisée pour combiner des variables et des chaînes littérales, ce qui est très pratique pour générer du contenu dynamique, comme des messages ou des URL.  
Voici trois façons de concaténer des chaînes en JavaScript :

## 1. Concaténation avec l'opérateur `+`

L'opérateur `+` est souvent le moyen le plus simple pour concaténer des chaînes en JavaScript. Son usage est intuitif, ce qui en fait un choix populaire pour combiner quelques chaînes rapidement.

```javascript  
const prenom = "Alice";  
const message = "Bonjour, " + prenom + "!";  
console.log(message); // Affiche : "Bonjour, Alice!"  
```  
Points importants à retenir :

- Si vous utilisez des variables ou des expressions dans votre concaténation, chacune d'elles doit être séparée par un opérateur `+`.  
- Cette méthode est simple mais peu lisible lorsque la chaîne devient plus complexe.  
- Quand on concatène plusieurs variables et valeurs littérales, cela peut vite devenir lourd.

## 4. Les Fonctions

Les fonctions sont des blocs de code conçus pour exécuter une tâche spécifique. Elles prennent des paramètres en entrée et peuvent retourner une valeur.  
  
- Déclaration d'une fonction :  
  
 function saluer(nom) {  
 return "Bonjour, " + nom;  
 }  
  
- Appel d'une fonction:  
  
 let message = saluer("Alice"); *// Renvoie "Bonjour, Alice"*  
  
  
Une fonction est utile car elle permet de réutiliser un même bloc de code plusieurs fois.

## 5. Les Structures de Contrôle

Les structures de contrôle permettent de contrôler le flux de votre programme selon des conditions spécifiques.  
  
- **if...else :** Permet de vérifier si une condition est vraie ou fausse et d'exécuter des instructions en conséquence.  
  
 **let** age = 20;  
 **if** (age >= 18) {  
 console.log("Vous êtes majeur.");  
 } **else** {  
 console.log("Vous êtes mineur.");  
 }  
  
  
- **switch**: Une autre structure de contrôle qui est utilisée lorsque vous avez plusieurs conditions à vérifier pour une seule variable.  
  
 let jour = "lundi";  
 switch (jour) {  
 case "lundi":  
 console.log("C'est le premier jour de la semaine.");  
 break;  
 case "vendredi":  
 console.log("Le week-end approche !");  
 break;  
 default:  
 console.log("Jour inconnu.");  
 }  
 ```

## 6. Les Boucles

Les boucles permettent d'exécuter le même bloc de code plusieurs fois. En JavaScript, on utilise principalement for et while.  
  
- **Boucle for:** Utile lorsque l'on sait à l'avance combien de fois il faut exécuter le code.  
  
 **for** (let i = 0; i < 5; i++) {  
 console.log(i); *// Affiche les nombres de 0 à 4*  
 }  
  
  
- **Boucle while:** Exécute le code tant qu'une condition est vraie.  
  
 let i = 0;  
 while (i < 5) {  
 console.log(i);  
 i++;  
 }  
 ```

## 8. Manipuler le DOM (Document Object Model)

Le DOM représente la structure d'une page HTML sous forme d'objet JavaScript, permettant aux développeurs de modifier le contenu et l’apparence de la page dynamiquement.  
  
Exemple :   
  
document.getElementById("titre").innerHTML = "Nouveau Titre";  
  
  
Dans cet exemple, on change le texte de l'élément avec l'ID `titre` en "Nouveau Titre".