Résumé des cours

11.11.2024

Session 01 : Fondamentaux du JS (1/2)

- Historique de javascript

Session 02 : Fondamentaux du JS (2/2)

Types et variables

1. Types de données et Variables

- **Types primitifs** : `Number`, `Boolean`, `String`, `Null`, `Undefined`
- **Variables** : `let`, `const`, et l'ancien `var`
- Convertisseurs de string : `parseint()` et `parseFloat()`
- **Opérations mathématiques** :

```
let x = 10 + 5; // addition
let y = x - 3;  // soustraction
console.log(y * 2); // multiplication et affichage du ré
sultat
```

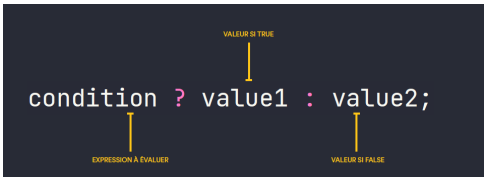
- **Exercice** : Déclare une variable `distance` et une variable `temps`, puis calcule la `vitesse` (distance / temps).

Comparaisons et conditionnelles

2. Opérateurs et Comparaisons

- **Comparaisons** : `==`, `===`, `<`, `>`, etc.
- Attribution d'une valeur `null` ou `undefined` : `??`
- **Logique booléenne** : Utilisation de `&&`, `||` et `!` pour les conditions complexes.

- Attribution d'une valeur conditionnelle à une variable : `?`



- **Exercice** : Écris un programme qui vérifie si une variable `âge` est un nombre valide et si elle représente un adulte (≥ 18).

3. Structures Conditionnelles

- **if...else** :

```
let age = 16;
if (age >= 18) {
  console.log("Accès autorisé");
} else {
  console.log("Accès refusé");
}
```

- **Opérateur ternaire** :

```
let message = (age >= 18) ? "Adulte" : "Mineur";
```

- Si on doit comparer une variable avec plein de variantes, privilégiez l'instruction switch
- **Exercice** : Écris un programme qui décide si une température donnée est « froide », « tempérée », ou « chaude » (> 25 °C).

Session 03 : Fonctions

Fonctions

6. Fonctions

- **Définition de fonction** :

```
function saluer(nom) {
  return `Bonjour, ${nom}`;
}
```

- Sans le return, la fonction retourne undefined
- **Fonction fléchée** :

```
const ajouter = (a, b) => a + b;
```

- **Exercice** : Crée une fonction `somme` qui prend un tableau en paramètre et retourne la somme des éléments.

Session 04 : Travailler avec des tableaux et des objets

Tableaux, objets et itération

4. Tableaux et Objets

- **Création de tableaux** :

```
let fruits = ["pomme", "banane", "cerise"];
```

- Stocker les tableaux dans une constante
- **Méthodes de tableaux** :
  - `Push` – ajouter à la fin
  - `Pop` – supprimer de la fin
  - `Shift` – Enlever du début
  - `Unshift` – Ajouter au début
  - `Concat` – Fusionne des tableaux
  - `Includes` – Cherche une valeur
  - `indexOf` – comme str.indexOf
  - `Join` – Créé une string à partir du tableau

- `Reverse` – Inverse un tableau
- `Slice` – copie une portion du tableau
- `Splice` – enlève / remplace des éléments
- `Sort` – classe notre tableau

- Connaître la longueur du tableau : `nomdutableau.length`
- Connaître un élément dans le tableau : `nomdutableau[n]`
- **Objets littéraux** :

```
const voiture = {
  marque: "Toyota",
  modèle: "Corolla",
  année: 2020
};
```

- **Exercice** : Crée un tableau de notes, puis calcule et affiche la moyenne.

5. Boucles et Itérations

- **Boucles classiques** :

```
for (let i = 0; i < fruits.length; i++) {
  console.log(fruits[i]);
}

while(j <= 10) {
    console.log(j);
    j++;
}
```

- **Boucle for...of pour tableaux** :

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

- **Exercice** : Écris un programme qui utilise une boucle pour afficher tous les nombres pairs entre 1 et 20.

Session 05 : Fonctions avancées

11. Set et Map en JavaScript

11.1 Set

- Un **Set** est une collection d'éléments **uniques**. Cela signifie qu'il ne peut pas contenir de doublons.
- **Création d'un Set** :

```
const fruits = new Set(["pomme", "banane", "cerise"]);
fruits.add("pomme"); // Ignoré car "pomme" existe déjà
console.log(fruits); // Set(3) { "pomme", "banane", "cer
ise" }
```

- **Méthodes principales** :
  - `add()` : ajoute un élément (si l'élément n'existe pas déjà).
  - `delete()` : retire un élément du Set.
  - `has()` : vérifie si un élément est présent.
  - `size` : retourne le nombre d'éléments dans le Set.
  - `clear()` : vide le Set.

```
const numeros = new Set();
numeros.add(5);
console.log(numeros.has(5)); // true
numeros.delete(5);
console.log(numeros.has(5)); // false
```

- **Itération avec for...of** :

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

- **Exercice** : Crée un Set avec une liste d'étudiants. Ajoute de nouveaux noms et supprime certains noms, puis affiche le Set.

11.2 Map

- Une **Map** est une collection de paires clé-valeur. Contrairement aux objets, une Map peut avoir des clés de n'importe quel type (y compris des objets ou des fonctions).
- **Création d'une Map** :

```
const capitales = new Map();
capitales.set("France", "Paris");
capitales.set("Suisse", "Berne");
console.log(capitales.get("France")); // "Paris"
```

- **Méthodes principales** :
  - `set(clé, valeur)` : ajoute ou met à jour une paire clé-valeur.
  - `get(clé)` : récupère la valeur associée à la clé.
  - `has(clé)` : vérifie si la clé existe.
  - `delete(clé)` : supprime une paire clé-valeur.
  - `size` : retourne le nombre d'éléments dans la Map.
  - `clear()` : vide la Map.

```
const restau = new Map();
restau.set("nom", "Chez Moi");
restau.set("ouverture", 9);
restau.set("fermeture", 22);

console.log(restau.get("nom")); // "Chez Moi"
restau.delete("ouverture");
console.log(restau.has("ouverture")); // false
```

- **Itération avec for...of** :

```
for (const [clé, valeur] of capitales) {
  console.log(`${clé} : ${valeur}`);
}
```

- **Exercice** : Crée une Map pour stocker les noms de cours et leurs codes. Ajoute, récupère et supprime des cours de la Map, puis affiche-la.

Session 06 - 07 : Manipuler le DOM

Introduction au DOM

DOM (Document Object Model) et Manipulation

- **Accéder aux éléments** :

```
const titre = document.querySelector("h1");
titre.textContent = "Bonjour Monde";
```

- La méthode `querySelector()` nous permet d'aller piocher un élément dans le DOM et `querySelectorAll()` permet d'aller piocher plusieurs choses.
- class : *.quelquechose*
- id : *#quelquechose*
- input type ="" : input
-
- **Modifier des styles** :

```
titre.style.color = "blue";
```

- **Exercice** : Modifie le texte d'un paragraphe HTML en utilisant JavaScript.

Session 08 : Événements

8. Écouteurs d'Événements

- **Ajouter un événement** :

```
const bouton = document.querySelector("button");
bouton.addEventListener("click", () => {
```

```
  console.log("Bouton cliqué !");
});
```

- **Propagation d'événements** (capture et bouillonnement).
- **Exercice** : Ajoute un événement sur un bouton pour changer la couleur de fond de la page.