Par Robert DIASSÉ

Le DOM en JavaScript

Introduction au DOM

Qu'est-ce que le DOM ?

Le DOM, ou Document Object Model (Modèle d'Objet du Document), est une interface de programmation pour les documents HTML et XML. Il représente la page web de manière structurée sous forme d'un arbre d'objets que les langages de programmation, comme JavaScript, peuvent manipuler. Le DOM permet d'accéder, de modifier, de supprimer et d'ajouter des éléments à la structure d'un document.

Utilité du DOM

Le DOM est essentiel pour créer des pages web interactives et dynamiques. Il permet aux développeurs de :

- Accéder et modifier le contenu HTML et les attributs des éléments.
- Modifier la structure du document en ajoutant ou supprimant des éléments.
- Réagir aux événements déclenchés par les utilisateurs (clics, saisies de texte, etc.).
- Manipuler le style CSS pour changer l'apparence des éléments.

Spécifications du DOM

Le DOM est une norme définie par le W3C (World Wide Web Consortium). Les spécifications du DOM définissent l'API que les navigateurs doivent implémenter pour permettre l'interaction avec les documents. Les versions courantes incluent le DOM Level 1, Level 2 et Level 3, chacune apportant des améliorations et des fonctionnalités supplémentaires.

Structure du DOM

Le DOM représente un document HTML sous forme d'un arbre d'objets. Chaque élément HTML, attribut et texte est un objet du DOM, appelé "nœud".

Types de Nœuds

1. **Document** : Représente le document entier.
2. **Element** : Représente un élément HTML.
3. **Attribute** : Représente un attribut d'un élément HTML.

4. **Text** : Représente le contenu textuel d'un élément ou d'un attribut.
5. **Comment** : Représente un commentaire dans le document.
6. **DocumentFragment** : Représente une partie détachée du document.

Exemple de Structure du DOM

Considérons le document HTML suivant :

```
<!DOCTYPE html>
<html>
<head>
    <title>Exemple de DOM</title>
</head>
<body>
    <h1>Bienvenue</h1>
    <p id="paragraphe">Ceci est un paragraphe.</p>
</body>
</html>
```

L'arbre DOM correspondant serait :

```
Document
├── html
│   ├── head
│   │   └── title
│   └── body
│       ├── h1
│       └── p#paragraphe
```

Accéder au DOM avec JavaScript

Sélection d'Éléments

JavaScript fournit plusieurs méthodes pour accéder aux éléments du DOM :

1. `getElementById` : Sélectionne un élément par son identifiant.

   ```
   let paragraphe = document.getElementById("paragraphe");
   ```

2. `getElementsByClassName` : Sélectionne des éléments par leur classe.

   ```
   let elements = document.getElementsByClassName("maClasse");
   ```

3. `getElementsByTagName` : Sélectionne des éléments par leur nom de balise.

```
let paragraphes = document.getElementsByTagName("p");
```

4. `querySelector` : Sélectionne le premier élément correspondant à un sélecteur CSS.

```
let premierParagraphe = document.querySelector("p");
```

5. `querySelectorAll` : Sélectionne tous les éléments correspondant à un sélecteur CSS.

```
let tousLesParagraphes = document.querySelectorAll("p");
```

Manipulation du Contenu

Vous pouvez manipuler le contenu des éléments sélectionnés en utilisant les propriétés et méthodes du DOM.

Modifier le Texte

```
let titre = document.querySelector("h1");
titre.textContent = "Nouveau Titre";
```

Modifier le HTML

```
let paragraphe = document.getElementById("paragraphe");
paragraphe.innerHTML = "Nouveau <strong>contenu</strong>";
```

Manipulation des Attributs

Accéder et Modifier les Attributs

```
let paragraphe = document.getElementById("paragraphe");
let id = paragraphe.getAttribute("id");
paragraphe.setAttribute("class", "nouvelleClasse");
```

Ajouter et Supprimer des Attributs

```
paragraphe.setAttribute("title", "Ceci est un paragraphe");
paragraphe.removeAttribute("title");
```

Manipulation de la Structure du DOM

Ajouter des Éléments

```
let nouveauParagraphe = document.createElement("p");
nouveauParagraphe.textContent = "Paragraphe ajouté";
document.body.appendChild(nouveauParagraphe);
```

Insérer des Éléments

```
let nouveauParagraphe = document.createElement("p");
nouveauParagraphe.textContent = "Paragraphe inséré";

let premierParagraphe = document.querySelector("p");
document.body.insertBefore(nouveauParagraphe, premierParagraphe);
```

Supprimer des Éléments

```
let paragraphe = document.getElementById("paragraphe");
paragraphe.remove();
```

Manipulation du Style

Vous pouvez manipuler les styles CSS des éléments via JavaScript en utilisant la propriété `style`.

```
let paragraphe = document.getElementById("paragraphe");
paragraphe.style.color = "blue";
paragraphe.style.fontSize = "18px";
```

Classes CSS

Vous pouvez ajouter, supprimer et basculer des classes CSS sur les éléments.

```
let paragraphe = document.getElementById("paragraphe");

// Ajouter une classe
paragraphe.classList.add("nouvelleClasse");

// Supprimer une classe
paragraphe.classList.remove("nouvelleClasse");
```

```
// Basculer une classe
paragraphe.classList.toggle("active");
```

Exercices

Exercice 1 : Sélection et Modification

Sélectionnez l'élément avec l'ID "paragraphe" et changez son texte en "Texte modifié".

Exercice 2 : Ajout d'Élément

Ajoutez un nouveau paragraphe avec le texte "Nouveau paragraphe" à la fin du corps du document.

Exercice 3 : Manipulation des Attributs

Ajoutez un attribut `title` à l'élément avec l'ID "paragraphe" avec la valeur "Ceci est un paragraphe".

Exercice 4 : Manipulation du Style

Changez la couleur du texte de l'élément avec l'ID "paragraphe" en rouge.

Exercice 5 : Classes CSS

Ajoutez la classe "important" à l'élément avec l'ID "paragraphe".

En maîtrisant le DOM, vous serez en mesure de créer des pages web interactives et dynamiques. Le DOM offre une multitude de possibilités pour manipuler les éléments HTML, réagir aux actions de l'utilisateur et modifier l'apparence et le contenu de vos pages web en temps réel.