Javascript Manipulation du DOM

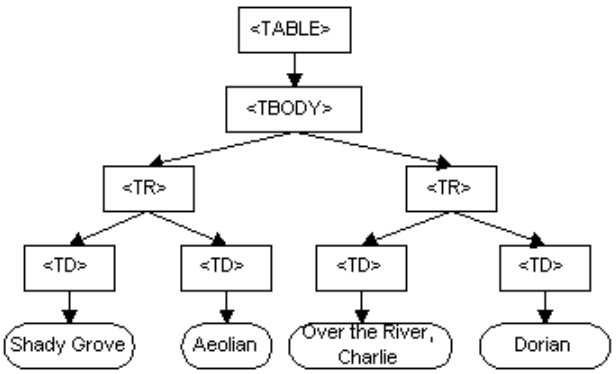
1. Présentation

Le Document Object Model (ou DOM) décrit une interface permettant à des scripts d'accéder ou de mettre à jour les éléments contenus dans une structure arborescente.

Avec le Modèle Objet de Document, les programmeurs peuvent construire des documents, naviguer dans leur structure, et ajouter, modifier, ou supprimer soit des éléments soit du contenu. Tout ce qui peut être trouvé dans un document HTML ou XML peut être accédé, changé, détruit, ou ajouté en utilisant le Modèle Objet de Document

Le modèle ne contient pas uniquement la structure du document mais également son comportement et celui des objets dont il est composé.

L'objet racine correspond à l'objet document.



2. La sélection des éléments

Par la valeur de l'attribut id	document.**getElementById**('nom') Retourne la balise dont l'attribut id a la valeur 'nom'.
Par le nom de l'élément (la balise)	document.**getElementsByTagName**('div') Renvoie dans un tableau toutes les balises let lesBalisesDiv = document.getElementsByTagName('div') Accès à la première balise div : let div1 = lesBalisesDiv[0]
Par la valeur de l'attribut class	document.**getElementsByClassName**('obligatoire') Renvoie dans un tableau les balises de type <.. **class**='obligatoire'>
Par la valeur de l'attribut name	document.**getElementsByName**('nom') Renvoie dans un tableau les balises de type <.. **name** ='nom'>
Par un sélecteur	document.**querySelector**('#avatar'); **Retourne le premier élément** du document qui correspond au groupe de sélecteurs passés en paramètre document.querySelector('input[name=choix]:checked').value; La case à cocher 'cochée' ayant un attribut name égal a 'choix'
Par un sélecteur	document.**querySelectorAll**(".boite"); Retourne dans un tableau **tous les éléments** de classe 'boite'
Par l'objet window (adressage indirect)	window['fenConsultation'].style.display = 'block'; accepte aussi en paramètre une variable qui contient la valeur d'un attribut id mais attention, la valeur de l'id doit commencer par une lettre let ligne = window['id' + id.value];

Les méthodes querySelector et querySelectorAll permettent l'utilisation des sélecteurs spécialisés et des pseudo-sélecteurs.

Champs dont l'id commence par 'message'	document.querySelectorAll('input[id^="message"]')
Champs dont l'id se termine par 'erreur'	document.querySelectorAll ('input[id$="erreur"]')
Champs dont l'id contient 'txt'	document.querySelectorAll ('input[id*="txt"]')

Il est possible d'accéder directement à un élément (input en général) sans passer par la méthode getElementById('nom') si le script ne contient pas de variable de même nom.

nom.value ='Dupont' équivalent à document.getElementById('nom').value = nom

3. Modifier le contenu d'une balise

Méthodes et Propriétés	Rôle
innerHTML="texte"	Remplace le contenu de l'objet par 'texte' avec interprétation des balises html contenues dans texte
textContent="texte"	Remplace le contenu de l'objet par 'texte' sans faire la moindre interprétation du texte.

Mettre à blanc toutes les balises dont l'id commence par message

```
document.querySelectorAll('[id^="message"]').forEach(element => element.innerText="");
```

4. Gérer les attributs d'une balise <nomBalise attribut='valeur' ….>

Notation element.methode ou element.propriete

Méthodes et Propriétés	Rôle
getAttribute('nom')	Renvoie la valeur de l'attribut 'nom' de l'objet. autre possibilité : element.nom
setAttribute('nom', valeur)	Ajoute un attribut 'nom' avec une valeur. autre possiblilite element.nom = valeur
removeAttribute(attribut)	Supprime l'attribut. Ex : btnSupprimer.removeAttribute('disabled')
dataset.id = valeur;	**Modifie la valeur de l'attribut 'data-id'. Concerne tous les attributs commençant par 'data-'**
attributes	Renvoie un tableau de tous les attributs de cet élément.
style	Objet contenant les attributs CSS de l'élément
onclick=	Ajoute une fonction de traitement sur l'événement click de l'élément. La valeur peut être le nom d'une fonction ou une fonction anonyme function() {…}
value = "texte"	Remplace la valeur de l'attribut value de l'objet (select par exemple)
className = " "	Modifie la valeur de l'attribut class de l'objet Préféré l'usage de la propriété classList (cf. 6.)

5. La propriété style

Notation : element.style.propriété

Propriétés (exemple)	Rôle
background	Définir la couleur de fond de l'objet
visibility="hidden|visible"	Masque ou affiche l'objet en le conservant dans le flux
display="none|block"	Masque ou affiche l'objet en le supprimant flux
style.color	Modifie la couleur du texte
fontSize	Taille de la police

```
let element = document.getElementById("element");
element.style.backgroundColor = "#ff0";
element.style.color = "#0f0";
element.style.fontSize = "1.2em";
element.style.textAlign = "center";
element.style.display = 'none';
```

6. La propriété classList

La propriété classList permet de gérer le contenu de l'attribut class d'une balise. Techniquement elle contient chaque classe dans une collection et offre les méthodes classiques d'une collection :

Méthode	Rôle
add(nom [,nom])	Ajoute la classe CSS nom dans l'attribut class de l'élément.
remove(nom [,nom])	Supprime la classe CSS nom dans l'attribut class de l'élément.
contains(nom)	Retourne vraie si nom se trouve dans la collection.
replace(a, b)	Remplace la classe a si elle existe par la classe b.
toggle(nom)	Permute (ajoute ou retire) la classe.
item(unIndex)	Retourne la classe située à l'index unIndex dans la collection
toString()	Retourne les valeurs de la collection dans une chaîne de caractères
length	Retourne le nombre d'élément dans la collection

Exemple : matricule.classList.add('erreur');

7. Comment se déplacer dans le DOM à partir d'un élément

Méthodes et Propriétés	Rôle
parentElement	Renvoie l'élément parent.
hasChildNodes()	Vérifie si un objet possède des nœuds enfants (childNodes), et Renvoie true si c'est le cas.
childNodes ou children	Renvoie un tableau de tous les nœuds à l'intérieur du nœud actuel
firstChild	Premier nœud = childNodes[0]
firstElementChild	Premier nœud (en ignorant le nœud texte et commentaire)
lastChild	Dernier nœud childNodes[this.childNodes.length-1].
nodeName	Renvoie le nom du nœud (le nom de l'élément HTML).
nodeType	Renvoie le type du nœud : 1 élément, 2 attribut, et 3 texte.
nodeValue	Renvoie ou fixe la valeur du nœud. Cette valeur est du texte si le nœud est un nœud textuel, l'attribut lui-même si c'est un attribut, ou 'null' (valeur nulle) si c'est un élément.
nextSibling	Retourne le nœud suivant (y compris les noeud texte et commentaire)
nextElementSibling	Retourne l'élément suivant (de même niveau)
previousElementSibling	Retourne le nœud précédent (de même niveau)
parentNode	Retourne le nœud parent

Rappel : la propriété length appliqué sur un tableau retourne le nombre d'élément du tableau

Exemple : stocker dans un tableau les valeurs de balises 'input' placées dans des balises td
Les balises tr sont encapsulées dans la div 'lesLignes'. Elles contiennent 2 balises td.

```
for(const tr of lesLignes.querySelectorAll("tr")) {
   let lesCellules = tr.childNodes;
   lesTaches.push({libelle : lesCellules[1].firstChild.value, duree : lesCellules[2].firstChild.value});
}
```

Recopier la valeur de l'input 'nom' dans l'input situé dans la troisième colonne de la balise tr dont l'id est stocké dans la variable id

```
document.getElementById(id).childNodes[2].firstChild.value  = nom.value;
```

8. Comment ajouter, modifier, supprimer des éléments dans le DOM

Pour ajouter un élément, il faut d'abord le créer en mémoire puis l'insérer dans le document à l'endroit souhaité.
Il existe plusieurs méthodes pour créer un élément :

Méthodes	Rôle
document.createElement('element')	Crée un nouvel élément (une balise)
document.createTextNode('chaine')	Crée un nouveau nœud de texte
document.cloneNode()	Copie le nœud entier avec tous les nœuds enfants.

Deux méthodes permettent d'insérer ce nouvel élément

Méthodes	Rôle
elt.appendChild(enfant)	Ajoute enfant en tant que dernier élément de elt.
elt.insertBefore(enfant, suivant)	Insère enfant avant l'élément suivant de elt. Si suivant n'est pas précisé enfant est ajouté comme dernier enfant

Il n'existe pas de méthode pour ajouter l'enfant au début de son conteneur mais il n'est pas difficile de la faire :

```
elt.insertBefore(enfant, elt.firstChild)
```

La suppression s'effectue par la méthode suivante

Méthodes	Rôle
elt.removeChild(enfant)	Supprime le nœud enfant dans les enfants de l'objet.

JavaScript ne permet pas de supprimer directement une balise. Pour le faire, il faut revenir sur le père et la balise à supprimer doit posséder un attribut id permettant de la repérer dans le DOM.

```
let element = document.getElementById(id);
element.parentNode.removeChild(element);
```

Dans le cadre d'une balise <table> il est possible de supprimer une balise <tr> en utilisant la méthode deleteRow(index) si on connait l'index de la ligne, cependant c'est rarement le cas et le fait de supprimer une ligne renumérote les lignes suivantes. Il est alors plus simple d'associer un id à chaque balise tr et de supprimer la balise tr

Exemple : Supprimer la ligne tr contenant une cellule td contenant une balise img accessible par this :

```
let tr = this.parentNode.parentNode;
tr.parentNode.removeChild(tr);
```

Pour effacer le contenu d'une balise il faut parcourir l'ensemble des enfants et les supprimer :

```
while (unNoeud.firstChild) unNoeude.removeChild(unNoeud.firstChild);
```

Une autre solution plus expéditive : **unNoeud.innerHTML=""**

Il est aussi possible de remplacer un élément par un autre en utilisant la méthode replaceChild (nouvelEnfant, ancienEnfant)

9. L'objet document

Méthodes et Propriétés	Rôle
document.location.href	Contient l'url de la page et permet donc de changer d'url document.location.href = "relance.php?id=" + id;
document.location.search	retourne les paramètres de l'url à partir de ? ➔?id=" + id;
document.write()	Permet d'écrire dans le document
document.forms[]	Tableau contenant l'ensemble des formulaires
document.img[]	Tableau contenant l'ensemble des images

10. Quelques exemples

- Générer une zone de liste

```
let id = document.createElement("select");
id.classList.add('form-control');
input.classList.add('m-2');
for (let motif of lesMotifs) {
   input.add(new Option(motif.libelle, motif.id));
}
```

- Récupérer les paramètres passés dans l'url ex: getcourse.php?numCourse=2

```
let lesParametresUrl = new URLSearchParams(document.location.search.substring(1));
numCourse = lesParametresUrl.get("numCourse");
```

- Modifier l'attribut src si la valeur n'existe pas en la remplaçant par l'image par défaut

```
let img = new Image();
img.src = 'avatar/' + leMembre.photo;
img.onerror = function () { img.src = 'avatar/0.png' }
```

- Ne rien afficher si l'image par défaut a aussi disparu ! : On masque la div pour en afficher une autre

```
img.onerror = function () {
      if (img.src.substr(-5) === '0.png') {
         aide.style.display = 'block';
         photo.style.display = 'none';
      } else {
         img.src = '../../avatar/0.png'
      }
   };
```

- Générer un tableau de deux colonnes (code et libelle) alimenter à partir d'un objet JSON 'lesLignes' à deux propriétés (code et libelle)

```
let lesCompetences = data;
let table = document.createElement('table');
let lesEntetes = ['Code', 'Libellé'];
let lesTailles = [100, 250];
let lesAlignements = ['center', 'left'];
let nbColonne = lesEntetes.length;
let tr = document.createElement('tr');
for(i = 0; i < nbColonne; i++){
    let td = document.createElement('td');
    td.setAttribute('style', 'border-style:solid; text-align:' + lesAlignements[i] + '; width:' +
lesTailles[i] + 'px');
    td.innerText =  lesEntetes[i];
    table.appendChild(tr);
}
for (let i in lesLignes) {
    let tr = document.createElement('tr');
    let id = lesLignes[i].code;
    let libelle = lesLignes[i].libelle;
    let td = document.createElement('td');
   td.setAttribute('style', 'border-style:solid; text-align:center');
    td.innerText = id;
    tr.appendChild(td);
    td = document.createElement('td');
    td.setAttribute('style', 'border-style:solid; text-align:left');
    td.innerText = libelle;
    tr.appendChild(td);
    table.appendChild(tr);
}
document.getElementById("idZone").innerHTML = "";
document.getElementById("idZone").appendChild(table);
```

- Supprimer une ligne possédant un id contenu dans une variable id

```
let ligne = document.getElementById(id);
ligne.parentNode.removeChild(ligne);
```

- Modifier le contenu d'une cellule d'une ligne possédant un id contenu dans une variable id

```
let ligne = document.getElementById(id);
let lesCellules = ligne.childNodes;
lesCellules[n].innerText = valeur
```

n représente le numéro de la colonne à modifier