LE DOM : DOCUMENT OBJECT MODEL

QU'EST-CE QUE LE DOM

Le ***Document Object Model*** ou **DOM** (pour modèle objet de document) est une interface de programmation pour les documents HTML, XML et SVG. Il fournit une représentation structurée du document sous forme d'un arbre et définit la façon dont la structure peut être manipulée par les programmes (code JavaScript), en termes de style et de contenu.

EXEMPLE D'UNE PAGE HTML

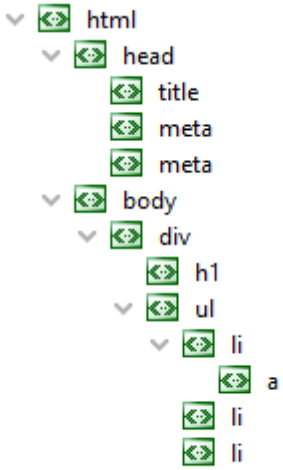
Le DOM se base sur la hiérarchie des objets contenue dans une page.

Le DOM représente le document comme un ensemble de nœuds et d'objets possédant des propriétés et des méthodes

```
<!DOCTYPE html>
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    </head>
    <body>
        <div><h1>Titre</h1>

            <ul>
                <li><a>first</a></li>
                <li>second</li>
                <li>third</li>
            </ul>

        </div>
    </body>
</html>
```



L'ensemble des nœuds est accessible par l'objet « document » en JavaScript.

L'objet document possède une propriété « childNodes » qui correspond à une liste de nœuds du « document ».

Dans la page HTML actuelle, le code suivant « document.childNodes » retourne la liste suivante de 2 nœuds :

```
NodeList(2) [<!DOCTYPE html>, html]
```

Le code suivant permet de récupérer la liste des nœuds du nœud « html »

```
var h = document.childNodes[1];
//ici h est un nœud représentant la balise HTML. Ce nœud contient lui-même d'autres nœuds.

var b= h.childNodes[1];  //b correspond au body
Var h1 = b.childNodes[0].childNodes[0];      //h1 correspond a la balise h1
```

Avec le DOM toute balise HTML peut être ainsi accessible via du code en JavaScript et devient modifiable.

Certains éléments comme les formulaires, le body et d'autres sont accessibles directement sans passer par childNodes qui peut s'avérer compliqué à utiliser.

Exemple :

```
Document.body.bgcolor= « red » ;
```

AJOUT D'ÉLÉMENT HTML EN JS

Il est possible ainsi d'ajouter des « nœuds » à volonté dans notre page. Le DOM propose plusieurs solutions pour cela. En voici une avec l'utilisation de createElement, et de appendChild.

```
var newElement = document.createElement("h1");              ➔ <h1></h1>
newElement.innerHTML="test de création d'éléments HTML");   ➔ <h1> test de création d'éléments HTML </h1>
document.body.appendChild(newElement);
```

MODIFICATION D'ATTRIBUT HTML

Avec le DOM, dès que l'on a accès à un élément HTML il est alors possible de modifier ses attributs.

Par exemple, nous souhaitons ajouter un href sur la balise <a> de notre page exemple :

```
<li><a>first</a></li>
```

Pour cela il faut commencer par avoir accès à cet élément <a>. C'est possible avec childNodes mais nous l'avons vu cela reste compliqué. Une solution existe et facile à mettre en œuvre en ajoutant un id à notre balise <a>

```
<li><a id="link1">first</a></li>
```

Cette balise devient maintenant accessible grâce à la fonction getEtlementByld de l'objet document.

```
var a=document.getElementById("link1");      //a pointe maintenant sur la balide <a>
a.href=http://www.google.fr                  //on modifie la propriété href
a.title="lien vers google"                   //on modifie l'attribut title
```

Comme l'attribut style est un attribut HTML il est aussi possible de modifier dynamiquement le style d'un élément HTML.

```
a.style.backgroundColor="red";
```

L'AJOUT D'ÉVÉNEMENT

Le JavaScript permet de réagir à des évènements. Ceux-ci peuvent être de différents types : clavier (key), souris (mouse), sur ouverture.

Exemple : lors de l'évènement click sur le body un message d'alert s'affiche.

```
document.body.onclick = function(){
    alert("coucou");
};
```

Un évènement (ou plusieurs) peut être ajouté à tous éléments HTML.