Chapitre 7 : Le DOM

DOM est l'acronyme de ***Document Object Model***. Autrement dit, les éléments d'une page html sont représentées sous la forme d'objets.

Les objets sont disposés en arbre. chaque nœud (éléments ou données textuelles).

```
monNoeud.nodeName;      // retourne le nom de la balise html
monNoeud.textcontent;   // si nodeName vaut "#text" les données sont situés ici
```

On peut demander à un nœud son arbre généalogique :

```
monNoeud.childNodes;    //retourne un tableau comprenant tous les fils d'un noeud
monNoeud.children;      // retourne un tableau comprenant tous les fils d'un noeud qui sont eux mêmes des éléments
```

7.1 Cibler un élément

7.1.1 Directement

Pour cibler un élément directement, il faut connaître son emplacement précis.

Exemple :

```
document.head
```

7.1.2 Indirectement

On peut cibler un élément à partir d'un autre élément

Exemple :

```
elementActuel.firstChild;               // premier enfant du noeud
elementActuel.firstElementChild;        // premier enfant qui est un élément
elementActuel.lastChild;                // dernier enfant du noeud
elementActuel.lastElementChild;         // dernier enfant qui est un élément
elementActuel.parentNode;               // dernier enfant qui est un élément
elementActuel.nextSibling;              // prochain frère au même niveau dans l'arbre
elementActuel.nextElementSibling;       // prochain frère au même niveau dans l'arbre qui est un élément
elementActuel.PreviousSibling;          // précédent frère au même niveau dans l'arbre
elementActuel.previousElementSibling;   // précédent frère au même niveau dans l'arbre qui est un élément
```

7.1.3 Suivant un critère

On peut rechercher un élément qui présente une caractéristique (exemple : un Id, une balise ou encore une classe)

Par ID

Retourne un élément

```
document.getElementById("monId");
```

Par Balise

Retourne une collection d'éléments

```
document.getElementsByTagName("h1");
```

> exemple de balises : "h1","p",...

Par Classe

Retourne une collection d'éléments

```
document.getElementsByClassName("maClasse");
```

Par Sélecteur CSS

```
document.querrySelector(".maclasse");   // retourne le premire élément
document.querrySelectorAll("#monId");   //retourne tous les éléments sélectionnés
```

7.2 Modifier un élément

Une fois un élément sélectionné, on peut vouloir le modifier.

7.2.1 Modifier son contenu

Code HTML

Modifier directement le code html de l'élément sélectionné :

```
element.innerHTML("CODE HTML")
```

Le texte dans l'élément

Modifier directement le code html de l'élément sélectionné :

```
element.textContent("Mon Texte Dans L'élément")
```

7.2.2 Modifier la valeur de ses attributs

Prenons l'exemple de l'attribut ***href*** suivant :

```
<a href="www.google.com">clique</a>
```

on peut le modifier grâce à :

```
element.href = "www.MonSite.com";
```

> attention cependant à ne pas oublier de sélectionner l'élément au préalable

7.2.3 Modifier son style

On peut aussi modifier la propriété style d'un élément :

```
element.style.backgroundColor = green ;
element.style.backgroundColor = white ;
```

7.2.4 Modifier sa classe

Les classes auxquelles appartiennent un élément sont stockées dans la propriété ***classList*** de l'élément. Le javascript donne des outils pour facilement gérer celles-ci :

```
element.classList.add("maClasse");      // ajoute une classe
element.classList.remove("maClasse");   // retire une classe
element.classList.toggle("maClasse");   // ajoute une classe si elle n'est pas présente et la retire si elle est présente
element.classList.contains("maClasse"); // booléen si la classe est présente
```

7.2.5 Modifier l'arborescence HTML

Ajouter des éléments

On peut créer de nouveaux éléments. Cependant, après les avoir crée, il faut les déplacer et les remplire

```
document.createElement("h1");
```

Retirer des éléments

Supprimer un élément ne peut se faire que depuis un noeud parent;

```
parent.removeChild(monNoeud);
element.innerHTML=""; //supprime tout
```

Modifier des éléments

```
element.appendChild(noeud);
element.insertBefore(noeud,autreNoeud); //on doit insérer avant l'autre noeud
element.remplaceChild(noeud,autreNoeud); //on remplace un noeud par un autre
```

7.3 Fenêtre (*window*)

L'objet ***window*** représente la fenêtre ou le document HTML est affiché. ***document*** est une des propriétés de l'objet ***window***.

Toutes les *fonctions globales* et les *variables globales* déclarées via ***var*** sont des propriétés de ***window***.

7.3.1 Quelques propriétés de window

```
window.document                 // document affiché
window.location                 // adresse de la page

window.history                  // historique de navigation pour cette fenêtre
window.history.length           // taille de l'historique de navigation
window.history.back()           //retourne à la page web prédécente
window.history.forward()        //va à la page web suivante

window.navigator                //informations sur le navigateur utilisé
window.navigator.appCodeName
window.navigator.appVersion
window.navigator.appName

window.screen                   //écran ou l'affichage se produit
window.screen.width
window.screen.height

/*DEPRECIATED*/ window.status   // texte d'état (barre de statut)
window.screenX                  // position sur l'écran
window.screenY
window.innerHeight              // taille interne/externe
window.outerHeight              // taille interne/externe
window.innerWidth               // taille interne/externe
window.outerWidth               // taille interne/externe
window.onload                   // action à exécuter dès que le contenu de la page est entièrement chargé (très pratique pour les initialisations !)
```

Méthodes de Window

```
window.alert();
window.prompt();
window.confirm();

window.moveTo(x,y);             //reposition de la fenêtre
window.moveBy(dx,dy);           //reposition de la fenêtre

window.resizeTo(x,y);           // ajuste la taille
window.resizeBy(dx,dy);         // ajuste la taille

window.scrollTo(x,y);           // déroulement
window.scrollby(dx,dy);         // déroulement
//  ces méthodes ne peuvent être utilisées que sur des fenêtres crééesvia Javascript (pour éviter les changements indésirables)

window.close();                 // ferme la fenêtre
window.open(url,nom,option);    // ouvre une nouvelle fenêtre (popup)
/*exemple*/
var fen2 = open("http://site.be", "Mon site", "resizable=no, location=no, width=200, height=100,menubar=no, status=no, scrollbars=no");

```

```
window.setTimeout(f,t[,param,param]) // exécute la fonction f dans t millisec et renvoie l'id de la tâche
window.clearTimeout(id)              // suspend l'exécution d'une tâche timeout

window.setInterval(f,t[,param,param]) // exécute la fonction f toutes les t millisec et renvoie l'id de la tâche
window.clearInterval(id)             //  suspend l'exécution d'une tâche interval
```