JAVASCRIPT: LE DOM

SOMMAIRE

WINDOW

L'objet window est l'objet global qui représente votre fenêtre. Tout le code JavaScript qui est exécuté sur une page a accès à cet objet.

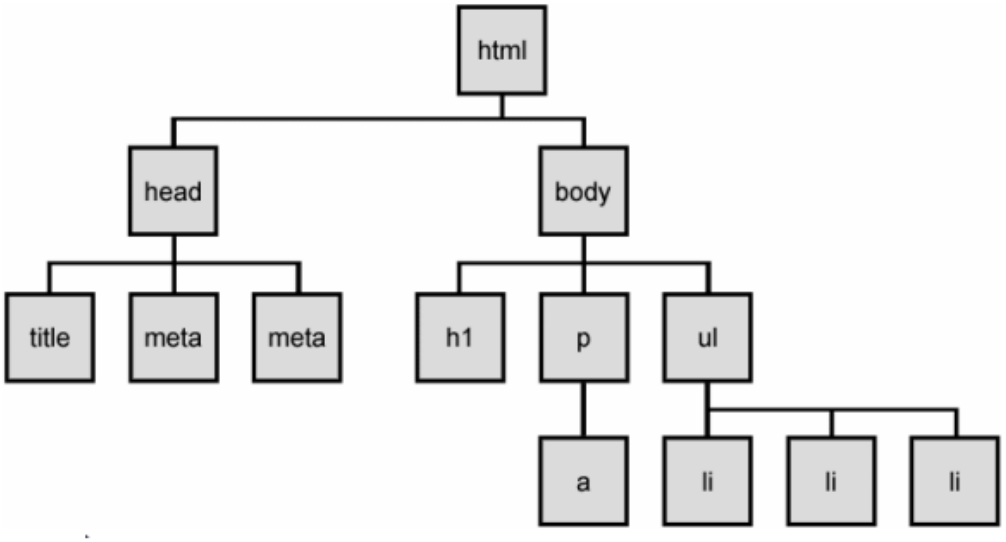
On peut avoir accès à des informations de l'historique window.history de l'onglet, ainsi que différentes informations comme la largeur window.innerWidth ou la hauteur window.innerHeight de la fenêtre.

Essayez un console.log(window) pour voir...

LE DOM

Le DOM est le document object Model, il est la structure du fichier HTML, il va nous permettre d'atteindre les différents éléments et de pouvoir agir dessus en javascript.

Sur le schéma ci-dessous, la base de notre DOM est la balise *HTML*. Celle-ci a 2 enfants, *HEAD* et *BODY* qui ont chacun des enfants également. Ici, BODY est le parent de *h1*, *p* et *ul*. Ces derniers, entres eux, sont frères.



CREONS NOTRE DOM

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="utf-8">
    <title>JavaScript</title>
    <link rel="stylesheet" href="css/style.css">
</head>
<body>
    <h2>Mon titre 1</h2>
    <div id="maPremiereDiv">
        Ceci est ma première div
    </div>

   <h2>Mon titre 2</h2>
   <div id="maDeuxiemeDiv">
        Ceci est ma deuxième div div
    </div>

    <h2>Mon titre 3</h2>
    <div id="maTroisiemeDiv">
        Ceci est ma troisième div
    </div>
    <p>
        <button id="buttonTitle">Modifier couleur de mon titre</button>
        <button id="changerBackground">Changer background div1</button>
    </p>
```

```
    <script src="js/script.js"></script>
</body>
</html>
```

LA SÉLECTION D'ÉLÉMENTS

Il existe plusieurs façons de sélectionner des éléments, en voici quelques unes :

GETELEMENTBYID

Cette methode permet de sélectionner un élément par son ID :

```
let divUne = document.getElementById("maPremiereDiv");
```

On peut alors agir dessus pour lui apporter des modifications :

```
divUne.style.backgroundColor = 'lightgrey';
divUne.style.padding = '1em';
```

Ici, on modifie la couleur de fond et le padding.

Remarque : les propriétés CSS qui sont habituellement en plusieurs mots comme background-color sont transformées en camelCase pour la manipulation en JS.

QUERYSELECTOR

Ce sélecteur est beaucoup plus souple, il permet de sélectionner n'importe quel élément du DOM avec un simple sélecteur CSS :

```
let divUne = document.querySelector("#maPremiereDiv");
```

Mais aussi avec des sélecteurs plus complexes :

```
let divUne = document.querySelector("div:nth-child(1)");
```

QUERYSELECTORALL

Permet de sélectionner plusieurs éléments en même temps

```
let divs = document.querySelectorAll("div");
```

Ici, nous sélectionnons toutes les div du document.

ADDEVENTLISTENER

Sert à ajouter un évènement à un élément du dom. Un évènement est une action de l'utilisateur ou du programme, par exemple un click, un appui sur une touche…

Liste des évènements :

http://devdocs.io/dom_events/

Syntaxe:

monElement.addEventListener('evenement',functionALancer) ;

DOMCONTENTLOADED

Il s'agit de l'évènement qui se produit lorsque la page web est chargée et qui indique que tous les éléments sont 'prêts' et qu'on peut agir sur la page avec le code javaScript.

```
document.addEventListener('DOMContentLoaded',function(){
    console.log('le dom est chargé');
})
```

EXEMPLE SUR DES BOUTONS

```
//partie1
function onClick2(){
    alert('coucou');
}


//partie2
function onClick1(){
    let titre = document.querySelectorAll('h2');
        for(i=0;i<titre.length;i++){
            titre[i].classList.toggle("blue");
        }
}

//partie3
function changerCouleur(){
    var maDiv=document.getElementById('maPremiereDiv');
    maDiv.classList.toggle("backgroundfonce");
}

document.addEventListener('DOMContentLoaded',function(){
    console.log('le dom est chargé');

    let button2=document.querySelector('#buttonTitle');
    let button1=document.getElementById('changerBackground');

    button2.addEventListener('click',onClick1);
    button1.addEventListener('click',changerCouleur);
```

```
})
```

REMOVEEVENTLISTENER

On peut mettre un évènement, on peut aussi l'enlever. Si à un moment donné, nous avons besoin de désactiver un bouton, il suffit de faire ceci :

```
button1.removeEventListener('click',changerCouleur);
```

Attention : il est important que la fonction de callback ainsi que l'évènement soient exactement les même qu'au moment de l'ajout de l'évènement.

CLASSLIST :

La propriété classList sert à gérer les classes de nos balises Html, elle est accessible et gérable avec javaScript.

Voici une liste de ses quelques-unes de ses méthodes :

- monElement.classList.add('maClasse') → Ajoute une classe
- monElement.classList.remove('maClasse')→Enlève une classe
- monElement.classList.toggle('maClasse')→ajoute ou enlève une classe
- monElement.classList.length →Retourne le nombre de classes que l'élément contient
- monElement.classList.contains('maClasse') →Retourne true si mon élément contient la classe ou false dans le cas contraire.

LA VARIABLE THIS

Dans un gestionnaire d'évènement, la variable this représente l'objet Dom qui a déclenché l'évènement.

Exemple :

On passe la souris sur une div de notre DOM, le background de celle-ci va changer de couleur :

On commence par sélectionner les divs et par leur affecter un évènement :

```
let divs = document.querySelectorAll('div');
    divs.addEventListener('mouseover',changerText);
```

Puis, on crée la fonction :

```
function changerText(){
    this.classList.add('green');
}
```

Au sein de la fonction, 'this' représentera uniquement la div sur laquelle nous avons passé la souris, celle qui a déclenché l'évènement, pas toutes les divs.

TEXTCONTENT

Modifie le contenu textuel d'un nœud du dom.

Exemple :

```
<div id= 'maDiv'>Hello Word</div>
```

Javascript:

```
let maDiv=document.getElementById('maDiv');
alert(maDiv.textContent);
```

→Renvoie "Hello Word"

```
madiv.textContent="Bonjour tout le monde"
```

→Modifie le contenu de la div

INNERHTML

On a vu le monElement.textContent, il existe son équivalent en HTML : monElement.innerHTML.

Contrairement à textContent, innerHTML va interpréter le code html compris dans la variable.

Exemple :

monElement.textContent= »Coucou » ; →Va retourner une valeur qui ne sera pas en gras alors que avec innerHTML si !