MÉMO4 - JS - CRÉER DE L'INTERACTION

Cours Javascript

Frédéric LOSSIGNOL

NB : Ce document sera complété par un prochain PDF où nous verrons notamment le format de données JSON, les fonctions de conversion et le localStorage.

Rappel

Nous avons vu précédemment les bases la programmations avec Javascript :

- les constantes, les variables
- la concaténation
- les types de variables et les conversions (parseInt() et parseFloat())
- les tableaux et les objets
- Les conditions
- Les boucles
- Les fonctions
- Des notions d'organisation du code en librairies de fonctions utiles

Maintenant nous allons aborder ce qui fait une des spécificités majeure de Javascript pour le web.

***JS** est conçu pour l'interaction utilisateur. C'*est donc le langage qui s'impose pour bâtir de l'interaction entre le visiteur et votre site web ou votre application.

Ce qui va suivre :

- **La sélection d' élements HTML**
- **La *modification d'une page web à travers le DOM***
- **La *gestion des évènements***

  (détecter le clic ou le survol de la souris sur un élément par exemple)

SOMMAIRE

1 Les gestionnaires d'évènements

Lien utile : https://developer.mozilla.org/fr/docs/Web/API/EventTarget/addEventListener

1.1 Pourquoi gérer les évènements utilisateur ?

Pour construire de l'interaction entre l'utilisateur et votre site ou votre application, tout simplement.

1.2 Les types d'évènements

Pour ne citer que les principaux, vous pouvez noter :

- *Le click (mousedown, click, mouseup)*
- *le double click (dblclick)*
- *une touche pressée (keydown, keypress, keyup)*
- *le survol (mousover, mouseout)*
- *Le chargement du DOM (DOMContentLoaded)*

Tester les types d'évènements sur le lien suivant :
https://jsfiddle.net/learnlab/ejhne43h/

1.3 La méthode addEventListener()

addEventListener() *permet l'assignation de gestionnaires d'événements sur un élément HTML cible. La cible d'un évènement peut être un nœud dans un document, le document lui-même, un élément window ou même un objet XMLHttpRequest.*

1.3 La syntaxe de addEventListener()

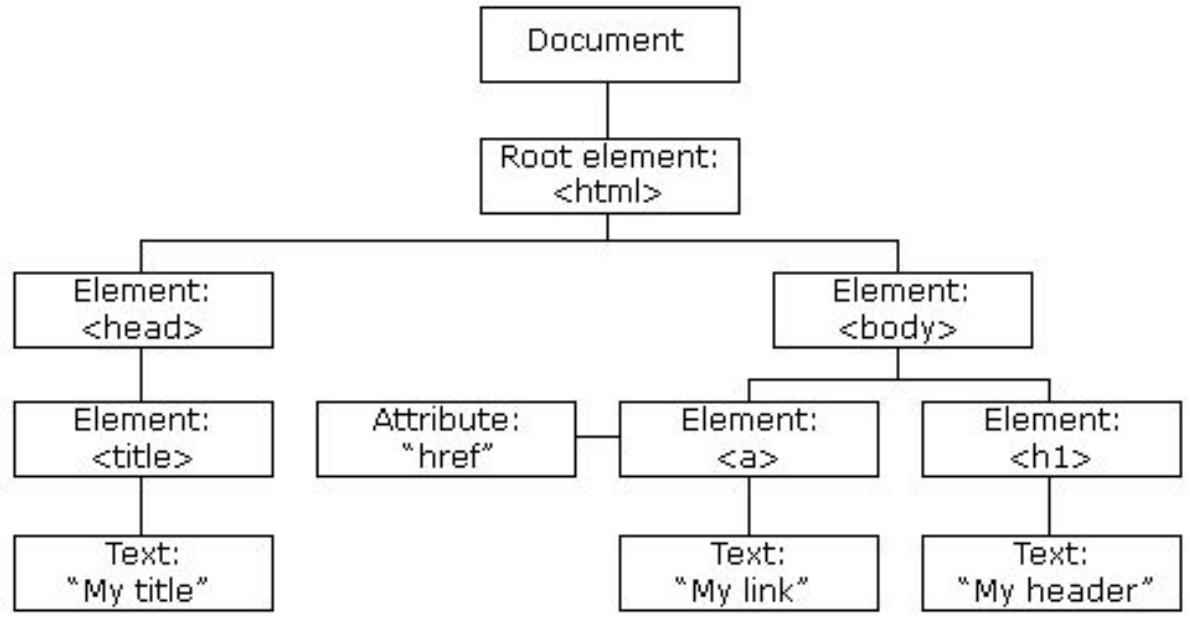
addEventListener(type, listener, useCapture)

- Type : (string) Représente le type d'événement à enregistrer (click, mousever, keyup, …)
- Listener : La fonction qui sera déclenchée à la détection de l'évènement

- UseCapture : (booléen)(facultatif) Déclenchement de l'événement pendant la phase de capture ou de propagation (capture si définit à true)

2 Le DOM (Document Object Model)

Le DOM est une représentation de votre page HTML sous forme d'arbre hiérarchique.



2.1 Qu'est-ce que le DOM et à quoi sert-il ?

A retenir :

le DOM est un standard du W3C qui décrit une interface indépendante de tout langage de programmation et de toute plate-forme, permettant à des programmes informatiques d'accéder ou de mettre à jour le contenu, la structure ou le style de documents HTML et XML.

2.2 Manipulation du DOM / Comment sélectionner un élément HTML en Javascript

Pour appliquer lire ou remplacer du contenu, il faut évidemment d'abord sélectionner un élément HTML cible sur lequel on veut faire cela. JS offre plusieurs façons de sélectionner un élément HTML, aussi appelé "noeud" (node en Anglais).

.getElementById('idCible')
Cible **l'élément** HTML qui a l'id idCible

.getElementsByClassName('classCible')
Cible **tous les éléments** HTML qui ont la class classCible et renvoie un tableau de ces éléments.

.getElementsByTagName('ul')
Cible **tous les éléments** HTML qui ont la balise ul et renvoie un tableau de ces éléments.

.querySelector()
Cible **le premier élément** du DOM indiqué en paramètre.

> Notez que querySelector() peut prendre en paramètre le nom d'une balise, une class ou un id et que vous devez, comme en CSS utiliser '.' pour désigner une classe et '#' pour désigner la classe ou l'id.

exemple *:*

```
// On sélectionne un élément et on le stocke dans une variable pour l'utiliser ensuite
var monElement;
monElement = document.querySelector('.brillant')
//  querySelector selectionne le 1erélément HTML qui a la classe brillant
```

2.3 Manipulation du DOM / Ajouter-supprimer ou remplacer du contenu

Via le DOM, vous pourrez en javascript ajouter, remplacer ou supprimer du contenu texte, du contenu HTML, des valeurs d'attributs, etc.

Quelques exemples courants de manipulation du DOM sur une page HTML :

.textContent = "" : *permet de lire ou remplacer du contenu texte à l'intérieur d'un élément HTML cible (sans lui assigner de valeur, il lira le contenu de l'élément cible).*

exemple *:*

```
// Supposons que nous avons dans notre document HTML : <h1 class='title'></h1>
var titre = document.querySelector('h1.title');
titre.textContent = "mon Super Titre" ;
// textContent injecte le texte "mon Super Titre" dans la balise <h1>
```

https://jsfiddle.net/learnlab/5qmLLwvs/ (exemple 1)

.innerHTML = "" : *permet d'ajouter ou remplacer du contenu HTML à l'intérieur d'un élément HTML cible (sans lui assigner de valeur, innerHTML lit le contenu de l'élément cible).* Note *:* Si vous souhaitez ajouter du contenu avec innerHTML sans supprimer le contenu existant, utiliser les signes d'assignation **+=** au lieu de **=**

exemple *:*

```
// Supposons que nous avons dans notre document HTML : <div class='contenu'></div>
var blocContenu = document.querySelector('.contenu');
var resume =  "Résumé : Dans cette incroyable métropole, chaque espèce animale cohabite avec les autres. Qu'on soit un immense éléphant ou une minuscule souris, tout le monde a sa place à Zootopia !"

blocContenu.innerHTML += "<p>"+ resume +"</p>" ;
// innerHTML ajoutera les balises <p></p> et le contenu de la variable resume dans la div.contenu
```

https://jsfiddle.net/learnlab/5qmLLwvs/ (exemple 2)

.getAttribute(nomDeLattribut) *: permet de lire le contenu de l'attribut de l'élément HTML cible (pratique pour récupérer le chemin d'une image par exemple).*

exemple *:*

```
// Supposons que nous avons dans notre document HTML :
// <img class='animals' src='images/perroquet.jpg'>
var imgElt = document.querySelector('.animals');
imgElt.getAttribute('src'); // Renverra 'images/perroquet.jpg'
```

.setAttribute(nomDeLattribut, valeur) *: permet de remplacer la valeur de l'attribut de l'élément HTML cible (utile par exemple dans un slider d'images si l'on souhaite changer dynamiquement la valeur de l'attribut src de la balise <img>.*

exemple *:*

```
// Supposons maintenant que nous souhaitons modifier dynamiquement la valeur de l'attribut src
imgElt.setAttribute('src', 'images/dog.jpg');
// Remplacement la source de l'image <img class='animals' src='images/dog.jpg'>
```

https://jsfiddle.net/learnlab/5qmLLwvs/ (exemple 3 et 4)

.dataset.nomDeData *: permet de lire une valeur ou de donner une nouvelle valeur à un attribut "data-NomDeData" d'une balise HTML*

exemple *:*

```
// Supposons que nous avons dans notre document HTML :
// <a class="name" href="#" data-email="arnaud@gmail.com">Arnaud</a>


var monElement = document.querySelector('.name');
monElement.dataset.email;
//  lit la valeur de data-email de notre balise<a>
monElement.dataset.email = 'freddy@orange.fr ' ;
//  écrit une nouvelle valeur dans data-email
```

https://jsfiddle.net/learnlab/5qmLLwvs/ (exemple 5)

.classList.add(nomDeLaClass) *:*

permet d'ajouter une class à un élément HTML.

.classList.remove(nomDeLaClass) *:*

permet de supprimer une class d'un élément HTML

.classList.toggle(nomDeLaClass)

permet d'ajouter et supprimer alternativement une class à un élément HTML

.classList.contains(nomDeLaClass)

permet de lister les class d'un élément HTML (retourne TRUE ou FALSE)

exemple *:*

```
// Supposons maintenant que nous souhaitons ajouter dynamiquement la class .green
// à l'élément  div#monId
var elt = document.querySelector('#monId')
elt.classList.add('green');
```

https://jsfiddle.net/learnlab/5qmLLwvs/ (exemple 7)

.value *permet de lire la valeur d'un input de type text, textarea ou select*

exemple *:*

```
// Supposons que la page affiche <input id="prenom" name="firstaname">
// et que l'utilisateur ait entré "Christian"
var name = document.querySelector('input[name='firstanme']').value
// console.log(name); retournera "Christian"
```

.createElement() *permet de créer un nouvel élément HTML (ou noeud) dans le DOM*

.createTextNode() *permet de crée un nouvel élément Texte dans le DOM.*

.appendChild() *permet d'ajouter dynamiquement un élément HTML ou Texte comme dernier enfant d'un élément cible parent qu'on aura préalablement sélectionné*

https://jsfiddle.net/learnlab/5qmLLwvs/ (exemple 6)

```
// Insérer dynamiquement du contenu HTML avec createElement() et appenChild()
// Une alternative plus performante à innerHTML vu plus haut

var blocParagraph = document.createElement('p'); // on crée une balise <p></p>
blocParagraph.id = 'elegant-pitch'; // On lui ajoute l'id elegant-Pitch

var lienAlloCine = document.createElement('a'); // de la même façon on crée une balise <a>
lienAlloCine.href = 'http://www.allocine.fr/film/fichefilm_gen_cfilm=223207.html';

var leTexteDuPitch = document.createTextNode("Zootopia est une ville qui ne ressemble à aucune autre : seuls les animaux y habitent ! … ");

var leTexteDuLien = document.createTextNode("Voir la fiche du film sur Allocine !");
// A ce stade les éléments HTML p, a et l'élément texte sont créé mais pas encore affichés.

// Nous les affichons maintenant avec appendChild()
blocParagraph.appendChild(leTexteDuPitch); // On insère le texte dans la balise p
lienAlloCine.appendChild(leTexteDuLien); // On insère le texte du lien a

document.getElementById('pitch').appendChild(blocParagraph);
// On insère la balise p dans l'élément #pitch
document.getElementById('pitch').appendChild(lienAlloCine);
//  On insère la balise a dans l'élément #pitch
```

3 Les outils du développeur / liens utiles

Les outils du développeur :

- Un éditeur de texte ou un IDE (Sublime Text, Netbeans, PHPStorm,...)
- L'outil développement sur les navigateurs (Firebug sur FireFox et l'outil de développement sur Chrome, tout deux accessibles avec le raccourci F12)
- DevDocs (www.devdocs.io) [Le dictionnaire du développeur]

  Application mobile pour apprendre les bases avec des petits exercices progressifs (sympa dans le métro :))
- (AppStore et PlayStore et windowsStore).
- => https://play.google.com/store/apps/details?id=com.sololearn.javascript&hl=fr
- => https://itunes.apple.com/us/app/learn-javascript/id952738987?mt=8
- => https://www.microsoft.com/fr-fr/store/apps/learn-javascript/9wzdncrdj6b4
- => http://www.sololearn.com/Course/JavaScript

Conseil : révisez les notions apprises en relisant document et le précédent (SEM1 SEM2)

et PRATIQUEZ , c'est la meilleure voie.

Si vous avez des questions ou des remarques, je suis joignable par Email :
frederic.lossignol@gmail.com

Frédéric Lossignol, votre prof de Developpement