# JavaScript by Mistera :

## Les erreurs fréquentes :

* getElementsByTagName() : il y a un « s » à Elements

## Introduction :

* Tout d'abord, un langage de programmation est un langage qui permet aux développeurs d'écrire du code source qui sera analysé par l'ordinateur.
* Un développeur, ou un programmeur, est une personne qui développe des programmes. Ça peut être un professionnel (un ingénieur, un informaticien ou un analyste programmeur) ou bien un amateur.
* Le code source est écrit par le développeur. C'est un ensemble d'actions, appelées instructions, qui vont permettre de donner des ordres à l'ordinateur afin de faire fonctionner le programme.
* un langage de programmation permet d'écrire du code source qui sera analysé par l'ordinateur. Il existe **trois manières** d'utiliser du code source :
  + **Langage compilé** : le code source est donné à un programme appelé compilateur qui va lire le code source et le convertir dans un langage que l'ordinateur sera capable d'interpréter : c'est le langage binaire, fait de 0 et de 1. (C, C++,…)
  + **Langage précompilé** : ici, le code source est compilé partiellement, généralement dans un code plus simple à lire pour l'ordinateur, mais qui n'est pas encore du binaire. Ce code intermédiaire devra être lu par ce que l'on appelle une « machine virtuelle », qui exécutera ce code. (C#, Java,…)
  + **Langage interprété** : dans ce cas, il n'y a pas de compilation. Le code source reste tel quel, et si on veut exécuter ce code,on doit le fournir à un interpréteur qui se chargera de le lire et de réaliser les actions demandées. Les scripts sont majoritairement interprétés. (JS, Python,…)

Chaque navigateur possède un interpréteur Javascript, qui diffère selon le navigateur. Si vous utilisez Internet Explorer, son interpréteur Javascript s'appelle JScript (l'interpréteur de la version 9 s'appelle Chakra), celui de Mozilla Firefoxse nomme SpiderMonkey et celui de Google Chrome est V8.

* Rôle du JavaScript : permet de dynamiser une page HTML, en ajoutant des interactions avec l'utilisateur, des animations, de l'aide à la navigation, comme par exemple : afficher/masquer du texte, faire défiler des images, créer un diaporama avec un aperçu « en grand » des images, créer des infobulles.
* Le JavaScript est un langage dit client-side, c'est-à-dire que les scripts sont exécutés par le navigateur chez l'internaute (le client). Cela diffère des langages de scripts dits server-side (va s'occuper de « créer » la page Web qui sera envoyée au navigateur) qui sont exécutés par le serveur Web. (PHP)

#### **PRATIQUES :**

* Emplacement de code JS est à la fin de head ou à la fin de body dans un code HTML, entre les balises <script> …… </script>
* Ou externe, <script src="hello.js"></script>
* Les commentaires :
  + Une seule ligne // ceci est un commentaire
  + Plusieurs lignes /\* ceci est un commentaire \*/
* Afficher une boite de dialogue : alert('Hello world!');

[Appuyiez ici pour plus de pratique !](file:///C:\Users\USER\AppData\Roaming\Microsoft\01%20Révision%20JS) (ctrl+click)

# Manipulation DOM

## Theorie

Le **Document Object Model** (DOM) est une API pour les documents HTML.

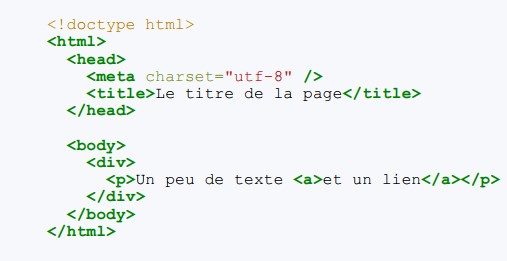
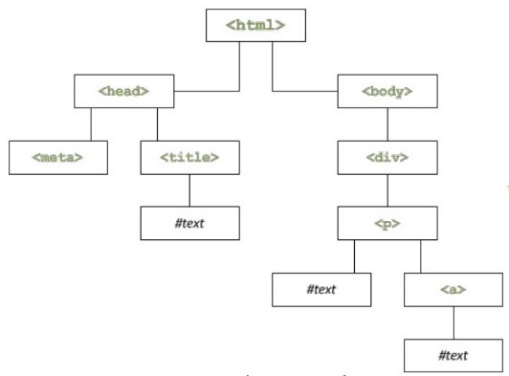
* Un **Application Programming Interface** (API) est un ensemble d'outils qui permettent de faire communiquer, entre eux, plusieurs programmes ou différents langages.
* grâce au DOM, nous allons pouvoir modifier des éléments HTML via JS(ou autre langage).
* Un **élément** = une paire de balises = un nœud.
* Propriété d’un élément = attribut de la balise (ex : href pour l’élément <a>)
* Les éléments HTML sont souvent composés d'attributs (l'attribut href d'un <a>), et d'un contenu(tout ce qui se trouve entre les ***balises ouvrante et fermante***).
* L'objet **window** est ce qu'on appelle un objet global, qui représente la fenêtre du navigateur. C'est à partir de window que le JavaScript est exécuté.
* l'objet window est dit *implicite*, c'est-à-dire qu'il n'y a généralement pas besoin de le spécifier :

windows.alert (« bonjour ») ; //**déconseiller**

est la même que :

alert (« bonjour ») ;

* lorsque vous déclarez une variable dans le contexte global de votre script, cette variable deviendra en vérité une propriété de l'objet window.
* toute variable non déclarée (donc utilisée sans jamais écrire le mot-clé var 🡺 **déconseiller**) deviendra immédiatement une propriété de l'objet window, et ce, quel que soit l'endroit où vous utilisez cette variable! (même dans une fonction, IEF,…) donc ces variables seront des variables globales.
* L'objet **document** est un sous-objet de window. Il représente la *page Web* et plus précisément la balise <html>.
* DOM pose comme concept que la page Web est vue comme un arbre, comme une hiérarchie d'éléments.



* L'objet document possède *trois méthodes* principales : getElementById(), getElementsByTagName() et getElementsByName().
  + **getElementById()**: méthode permet d'accéder (récupérer) à un élément en connaissant son attribut id

Exemple : <div id="myDiv">

code JavaScript : var div = document.getElementById('myDiv');

* + **getElementsByTagName()** : méthode permet de récupérer (sous la forme d'un tableau) tous les éléments par son *Tag Name*(nom de balise). Si, dans une page, on veut récupérer tous les <div>, il suffit de faire comme ceci :

var divs = document.getElementsByTagName('div');

* + **getElementsByName()** : méthode permet de récupérer les éléments qui possèdent un attribut *name* que vous spécifiez. L'attribut name n'est utilisé qu'au sein des formulaires, et est déconseillée dans tout autre élément que celui d'un formulaire. Par exemple, vous pouvez vous en servir pour un élément **<input>** mais pas pour un élément **<map>**.
* **Accéder aux éléments grâce aux technologies récentes :**
* **querySelector()** : renvoie le premier élément trouvé correspondant au sélecteur CSS mentionné.

let span = document.querySelector("#menu .monDiv span");//seulement le premier span inclus dans la class= « monDiv » inclus dans id= « menu ».

let span = document.querySelector("span"); //seulement le premier span dans le code HTML

* **querySelectorAll()** : renvoie tous les éléments (sous forme de tableau) correspondant au sélecteur CSS fourni.

let span = document.querySelector("#menu .monDiv span");//seulement le premier span inclus dans la class= « monDiv » inclus dans id= « menu ».

let span = document.querySelectorAll("span"); // tous les spen dans HTML

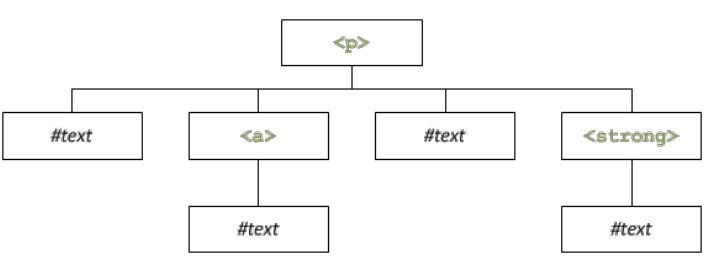
* élément HTML est un objet qui appartient à plusieurs objets, et de ce fait, qui hérite des propriétés et méthodes de ses objets parents.

Parent------------------------------------------------------🡪 sous-Objet

Node 🡪 Element 🡪 HTMLElement 🡪 HTMLDivElement

* Pour interagir avec les attributs, l'objet Element nous fournit *deux* méthodes, getAttribute() et setAttribute() permettant respectivement de *récupérer* et *d'éditer* un attribut.
* firstChild et lastChild servent respectivement à accéder au premier et au dernier enfant d'un nœud.



firstChild() lastChild()

firstElementChild() lastElementChild()

# LES EVENEMENT

* Les événements permettent de déclencher une fonction selon qu'une action s'est produite ou non. Par exemple, on peut faire apparaître une fenêtre alert() lorsque l'utilisateur survole un élément d'une page Web.
* Liste des événements

|  |  |
| --- | --- |
| Nom de l'événement | Action pour le déclencher |
| click | Cliquer (appuyer puis relâcher) sur l'élément |
| dblclick | Double-cliquer sur l'élément |
| mouseover | Faire entrer le curseur sur l'élément |
| mouseout | Faire sortir le curseur de l'élément |
| mousedown | Appuyer (sans relâcher) sur le bouton gauche de la souris sur l'élément |
| mouseup | Relâcher le bouton gauche de la souris sur l'élément |
| mousemove | Faire déplacer le curseur sur l'élément |
| keydown | Appuyer (sans relâcher) sur une touche de clavier sur l'élément |
| keyup | Relâcher une touche de clavier sur l'élément |
| keypress | Frapper (appuyer puis relâcher) une touche de clavier sur l'élément |
| focus | « Cibler » l'élément |
| blur | Annuler le « ciblage » de l'élément |
| change | Changer la valeur d'un élément spécifique aux formulaires (input, checkbox, etc.) |
| select | Sélectionner le contenu d'un champ de texte (input, textarea, etc.) |
| submit | Envoyer le formulaire |
| reset | Réinitialiser le formulaire |