

JavaScript 2^{ème} partie

Toujours selon [@MDN](#):

JavaScript (« JS ») est un langage de script léger, orienté objet. Le code JavaScript est **interprété ou compilé à la volée**. C'est un langage à **objets** disposant d'un **typage faible** et **dynamique**.

Rappel: Dans cet exemple on instancie une variable appelée `car` en lui affectant un objet, puis on lui affecte une chaîne de caractères.

```
let car = {  
  brand: "Reliant"  
  year: 1962  
};  
  
car = "Reliant";
```

Document Object Model (DOM)

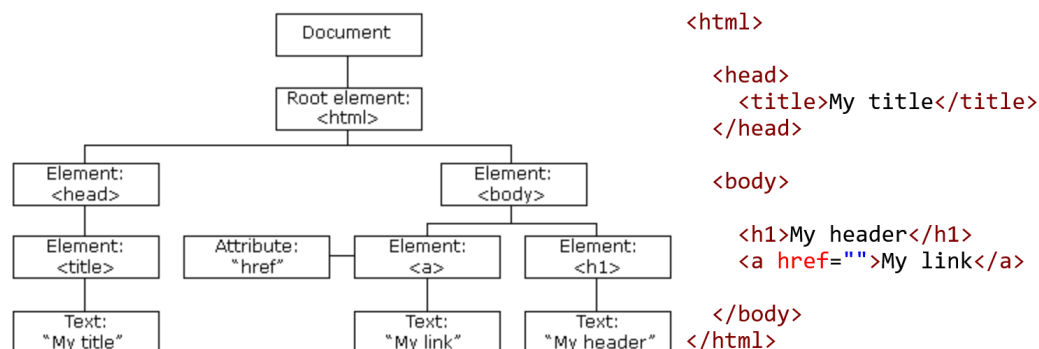
Le DOM est une interface de programmation (API) pour l'HTML, le XML et le SVG. Dans le cadre de ce cours, le DOM sert à "connecter" une page web au JavaScript.

Même si le DOM est essentiellement utilisé en JavaScript, ce n'est pas du JavaScript.

Le DOM doit être distingué du HTML initial, il peut être modifié par le navigateur.

Le DOM:

- fournit une représentation structurée d'un **document** en arbre
- expose des méthodes permettant d'accéder au document et d'y apporter des modifications dans sa **structure**, son **style** et son **contenu**.
- représente le document en **nœuds** et **objets** ayant chacun leurs propriétés
- Permet d'écouter et de gérer des **événements** sur les nœuds.



Voir `dom-01_document.html`

Modification du DOM

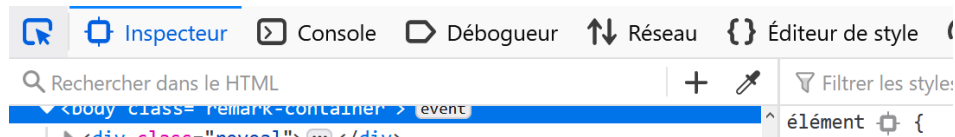
Le DOM peut-être modifié, par exemple en JavaScript:

```
document.getElementById("modification-du-dom").textContent="Le titre de cette page s'est modifié!";
```

- `getElementById` est une méthode de `Document` qui retourne un `Element`.
- `textContent` est un propriété de `Node` héritée par `Element`.

La console ou certains éditeurs de code proposent de l'autocomplétion de méthodes et propriétés de DOM.

Pour naviguer dans le DOM, rien de mieux l'**outil de sélection d'éléments** dans les outils de développement du navigateur!



Voir `dom-02_alert.js`

Sélecteurs les plus courants

Exemples	Signification
<code>.getElementById()</code>	Sélectionne l'élément avec l'id en paramètre
<code>.getElementsByTagName()</code>	Sélectionne tout les éléments selon le tag donné (par exemple 'p')
<code>.getElementsByName()</code>	Sélectionne tout les éléments selon l'attribut name
<code>.getElementsByClassName()</code>	Sélectionne tout les éléments selon ayant la classe donnée
<code>.children</code>	Sélectionne tout les enfants
<code>.textContent</code>	Sélectionne le texte à l'intérieur de l'élément
<code>.querySelector()</code>	Sélectionne des éléments selon la syntaxe CSS

- Les sélecteurs peuvent être chaînés, par ex:
`document.getElementById("s-lecteurs-les-plus-courants").children`
- L'autocomplétion dans la console vous aide à connaître les méthodes disponibles sur un élément

Gestion d'événements

Une page HTML se charge → le `document` se crée. Comment savoir quand la page a terminé de se charger?

```
// Création d'un objet qui a deux propriétés de type string
const myContent = {
  alertText: "Le document est chargé",
  alertLink: "https://developer.mozilla.org/fr/docs/web/API"
};

// Utilisation du DOM!
document.onreadystatechange = function() {
  console.log("L'état du document a changé:", document.readyState);
  if (document.readyState === "complete") {
    console.log(myContent.alertText);
  }
}
```

Quelques autres événements

Ouvrez la console avec `F12` pour voir ce qui s'y passe.

Plein d'autres événements existent:

<https://developer.mozilla.org/fr/docs/Web/Events>

Voir `dom-03_ready.js`

Le mot-clé `this` dans l'HTML

En JavaScript, `this` se réfère au contexte dans lequel on se trouve. [Voir le chapitre sur la portée des variables](#)

Utilisé comme argument dans le *callback* d'un *event*, il permet de passer l'élément actuel au *callback*:

Dans cet HTML, `doSomething()` est la fonction appelée quand `onkeyup` est fait. Autrement dit, `doSomething()` est le *callback* de `onkeyup`:

```
<input type="text" onkeyup="doSomething(this)">
```

En JavaScript

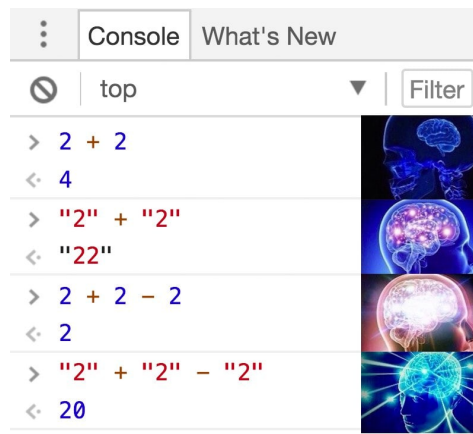
```
function doSomething(someElement) {
  console.log(typeof someElement); // object
  console.log(someElement.value); // le contenu que l'utilisateur tape dans l'input
}
```

Le mot clé `this` peut également être utilisé dans une fonction JavaScript mais cela sort du contexte de ce cours. Si vous souhaitez en savoir plus:

https://www.w3schools.com/js/js_this.asp

Exercices

Faites les exercices dom-01 à dom-04.

A screenshot of a web browser's console. At the top, there are tabs for 'Console' and 'What's New'. Below the tabs, there's a search bar with 'top' and a 'Filter' button. The console displays four lines of JavaScript code and their results: 1. '> 2 + 2' followed by '< 4'. 2. '> "2" + "2"' followed by '< "22"'. 3. '> 2 + 2 - 2' followed by '< 2'. 4. '> "2" + "2" - "2"' followed by '< 20'. To the right of the code, there are four small images of human brains with glowing neural activity.

Pour vous aider, cette page peut servir de "dictionnaire" de tags HTML avec chaque fois un détail des événements et attributs disponibles:

<https://www.w3schools.com/tags/default.asp>

Hors série: qu'est ce que le Path ?


`Path` est une variable d'environnement disponible sur les systèmes Windows, Linux et Unix. Elle permet de mettre au courant le système de l'existence d'un programme ainsi que de l'endroit où il se trouve. Une fois le système au courant, on pourra taper directement le nom dudit programme dans un terminal, sans avoir à taper son chemin complet.

Les programmes concernés dans ce cours sont `npm`, `python` et `psql`.

Sur windows, le `Path` existe à deux niveaux:

- Au niveau du système, pour tous les utilisateurs. Privilèges d'administration nécessaires pour le changer
- Au niveau de l'utilisateur courant.

Pour ajouter un programme au `Path` sur Windows:

1. Cliquer sur le menu démarrer ou presser la touche  (windows)
2. Taper "variables" et choisir "Modifier les variables d'environnement pour votre compte"
3. Cliquer sur `Path` puis "Modifier"
4. Ajouter le chemin du dossier contenant le programme à la fin de la liste.