

05.JavaScript & DOM

3 décembre 2025

Développement web il3

JavaScript & DOM

HE-Arc (DGR) 2025

JavaScript hier

- Page web = HTML (+ CSS + JavaScript)
- Exécuté par le browser (client)
- Interprété, faiblement typé, OO
- Historiquement
 - Depuis Netscape 2 (1995, Brendan Eich)
 - Petites applications exécutées par le navigateur
 - DHTML : rollovers, validation de formulaires, ...

JavaScript aujourd’hui

- Page web = HTML + CSS + **JavaScript**
- Compilation JIT
- HTML5, AJAX, bookmarklets
- Userscripts¹ : Tampermonkey², Brave Custom scriptlets³
- One Page Apps
- Implémentations hors-browser
 - Node.js, Spidermonkey, Rhino

1. <https://github.com/awesome-scripts/awesome-userscripts>

2. <https://www.tampermonkey.net/>

3. <https://brave.com/privacy-updates/32-custom-scriptlets/>

- script d'app (Qt, Notepad++, ...)
- Langage cible de compilateurs : emscripten⁴, WebAssembly⁵
- Embarqué : Espruino⁶, robotique : Node Bots⁷, CylonJS⁸
- Applications Desktop : Electron⁹, sciter¹⁰

*Script

- ECMAScript : Norme depuis 1997
 - Juin 2025 : ECMA-262 16th edition¹¹
 - Support¹² des différentes implémentations
 - Conversions avec BabelJS¹³
- JavaScript : implémentation Firefox (réf. MDN)
- Variantes (à transpiler) :
 - Typescript¹⁴ : variante fortement typée, avec des classes (MS)
 - Coffeescript¹⁵
 - sucre syntaxique
 - compilé -> js

JavaScript

- Différentes implémentations¹⁶ : navigateur, srv, apps, ...
- Permissif : du mauvais code est peu maintenable
 - Design Patterns¹⁷
 - Bonnes pratiques¹⁸ (src¹⁹)
 - Exploring JavaScript²⁰
- Interface pour scripter le navigateur

4. <https://emscripten.org/>

5. <http://webassembly.org/>

6. <http://www.espruino.com/>

7. <https://nodebots.io/>

8. <https://cylonjs.com/>

9. <https://electronjs.org/>

10. <https://sciter.com/>

11. <https://www.ecma-international.org/publications-and-standards/standards/ecma-262/>

12. <https://compat-table.github.io/compat-table/es2016plus/>

13. <https://babeljs.io/>

14. <https://www.typescriptlang.org/>

15. <http://coffeescript.org/>

16. https://en.wikipedia.org/wiki/List_of_ECMAScript_engines

17. <https://patterns.addy.ie/>

18. <https://web.archive.org/web/20250827211325/http://www.jstherightway.org/>

19. <https://github.com/jstherightway/js-the-right-way/>

20. <https://exploringjs.com/js/index.html>

- Accès et modification du contenu via DOM
- Bookmarklets ²¹, exemples ²²
- Requêtes HTTP (Fetch API, Xml Http Request)
- Développement d'applications complètes, parfois offline
- Langage de script généraliste (paquets npm)

Caractéristiques du langage

- Orienté Objet par prototype
- Syntaxe proche de C, Java
- Faiblement typé :
 - Pas de déclaration, type déterminé par la dernière affectation
 - Risque : typo => nouvelle variable. Utiliser const et let
- Types :
 - Primitifs : Boolean Null Undefined Number String Symbol
 - Objets : Object Function
- Particularités
 - Prototypes ²³
 - Fermetures ²⁴
 - Promesses ²⁵ (MDN ²⁶, Google ²⁷)

Fonctions

- Pas de type de retour
- Possibilité de retourner ou non une valeur
- Sans retour, valeur spéciale : undefined
- Pas de surcharge (la dernière définie prime)
- function est un type
- Fonctions imbriquées, anonymes
- Fonctions globales :

```
escape(), unescape(), isFinite(), isNaN(),
parseFloat(), parseInt(), Number(), String(),
eval(), ...
```

-
21. <http://www.howtogeek.com/125846/the-most-useful-bookmarklets-to-enhance-your-browsing-experience/>
 22. <http://www.hongkiat.com/blog/100-useful-bookmarklets-for-better-productivity-ultimate-list/>
 23. https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Le_mod%C3%A8le_objet_JavaScript_en_d%C3%A9tail
 24. http://www.w3schools.com/js/js_function_closures.asp
 25. <https://www.promisejs.org/>
 26. https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Promise
 27. <https://web.dev/articles/promises?hl=fr>

JavaScript dans la page web

- Éléments <script> exécutés dans l'ordre de la page
- Conseillé de les placer en fin de page ²⁸
- Evénements (onclick, onerror, onsubmit, ...)
 - Embarqués dans les balises (onXXX)

```
<div id="intro" onclick="change(); ">
```

Utiliser DOM

```
<script type="text/javascript">  
  
document.getElementById("intro").onclick = change;  
  
</script>
```

- Conseillé d'inclure le code (attribut src)

```
<script type="text/javascript" src="script02.js"></script>
```

language="JavaScript" est déprécié et type vaut par défaut text/javascript.

The type attribute gives the language of the script or format of the data. [...] The default, which is used if the attribute is absent, is “text/javascript”.

HTML5 : script ²⁹

Unobtrusive JS ³⁰

- Séparation JS...

```
document.addEventListener("DOMContentLoaded", function() {  
    document.getElementById('date').addEventListener("change", validateDate);  
};
```

- ...et HTML

```
<input type="text" name="date" id="date" />
```

- Dégradation élégante
 - Alternatives pour un browser ne supportant pas JS
 - Accessibilité

28. <https://stackoverflow.com/questions/1638670/javascript-at-bottom-top-of-web-page>

29. <https://www.w3.org/TR/html5/scripting-1.html#the-script-element>

30. https://en.wikipedia.org/wiki/Unobtrusive_JavaScript

- Les fonctionnalités restent accessibles en cas d'erreur
- Utilisabilité
 - Le script doit faire gagner du temps, pas distraire

It is an incredibly popular mistake to use `load` where `DOMContentLoaded` would be much more appropriate, so be cautious.

MDN : `DOMContentLoaded`³¹

Node.js³² / Deno³³

- Node.js : une implémentation hors navigateur
 - environnement d'exécution + bibliothèques
 - event driven, non-blocking IO -> scalable
 - V8 engine
 - scripts exécutables sans navigateur
 - npm³⁴ : gestionnaire de paquets
 - gulp : make js
- Exemples³⁵ d'applications
 - gulp, grunt, bower, yarn
 - browserify
 - serveur http
 - express, cordova, forever, dev, pm2, karma, sails, phantomjs
- Tuto³⁶, Playground³⁷

DOM

- Document Object Model
- Représentation arborescente de la page
- Accessible depuis objet JS document
- Possibilité d'accéder au contenu de la page :
 - Lecture
 - Modification
 - Ajout
- JS peut donc modifier le contenu d'une page

31. <https://developer.mozilla.org/en/docs/Web/Events/DOMContentLoaded>

32. <https://nodejs.org>

33. https://www.reddit.com/r/node/comments/nx9qr/deno_vs_nodejs_a_comparison_you_need_to_know/

34. <https://www.npmjs.com>

35. <https://colorlib.com/wp/npm-packages-node-js/>

36. <https://www.tutorialspoint.com/nodejs/index.htm>

37. <https://stackblitz.com/edit/node-js-playground?file=index.js>

DOM³⁸

```
<html>
<head>
    <title>My title</title>
</head>
<body>
    <h1>A heading</h1>
    <a href="#">Link text</a>
</body>
</html>
```

L'objet Document

- Trouver ou modifier des éléments
- Méthodes de Document

```
querySelector(), querySelectorAll(),
getElementById(), getElementsByTagName(), getElementsByClassName(),
createElement(), createTextNode()
```

- Méthodes de Node (appel depuis nœud parent)

```
insertBefore(child), appendChild(child),
removeChild(child), replaceChild(new,old)
```

Ajouter un noeud

```
function addNode() {
    var inText = document.getElementById("textArea").value;
    var newText = document.createTextNode(inText);

    var newGraf = document.createElement("p");
    newGraf.appendChild(newText);

    var docBody = document.getElementsByTagName("body")[0];
    docBody.appendChild(newGraf);
}
```

- Création du nouveau nœud :

38. <http://bioub.github.io/dom-visualizer/>

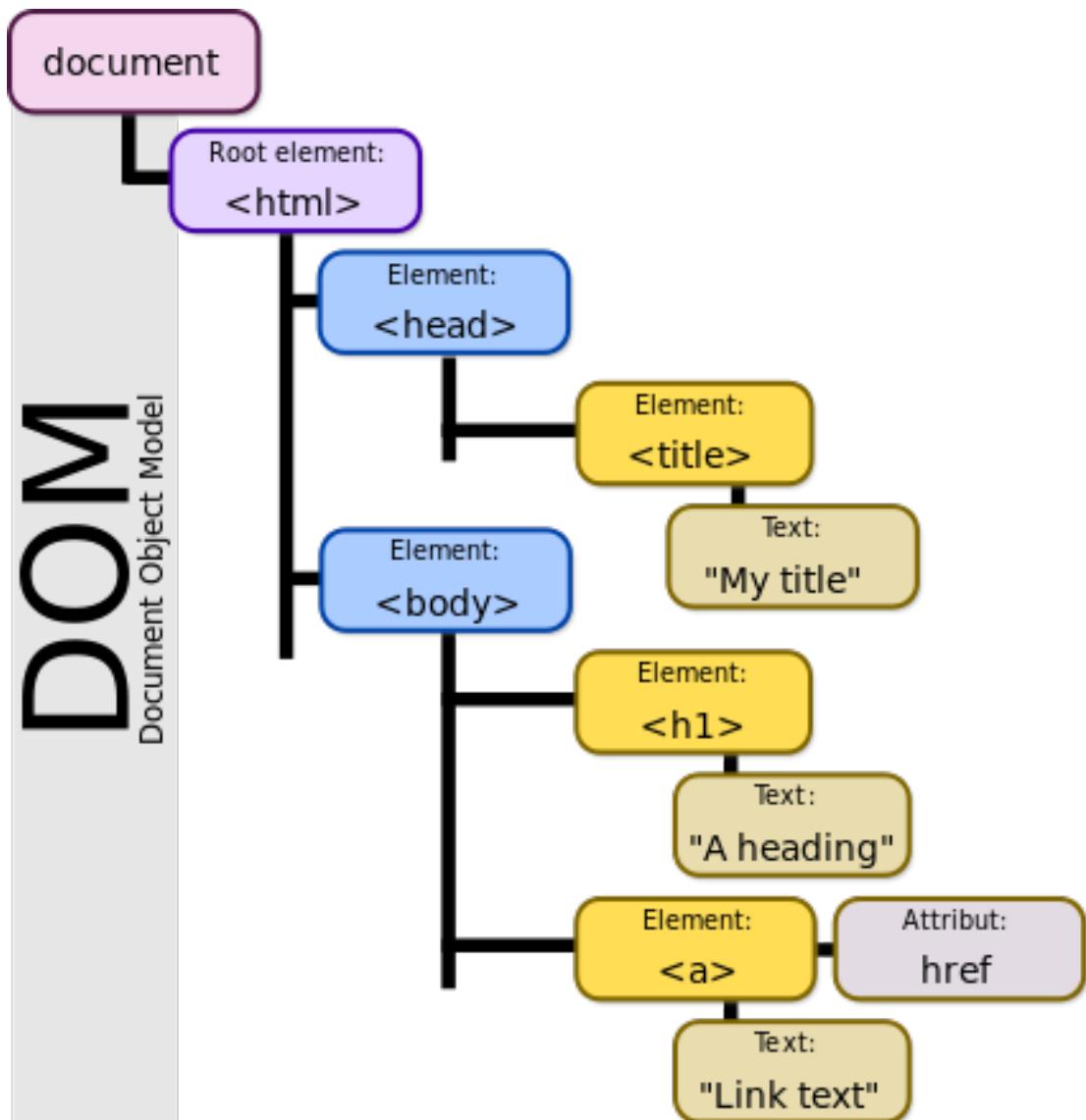


FIGURE 1 – DOM tree

- newText contient le texte à ajouter
- newGraf est un élément p qui contient le texte
- Ajout du nœud comme une feuille de body :
 - Sélection du parent (le premier noeud body)
 - Ajout du nouveau nœud depuis son parent

Supprimer un nœud

```
function delNode() {
  var allGrafs = document.getElementsByTagName("p");

  if (allGrafs.length > 1) {
    var lastGraf = allGrafs.item(allGrafs.length-1);
    lastGraf.parentNode.removeChild(lastGraf);
  }
  else {
    console.error("Nothing to remove!");
  }
}
```

- Sélection du nœud à supprimer :
 - allGrafs contient tous les éléments p
 - lastGraf contient le denier du tableau allGrafs
- Suppression :
 - Suppression du nœud sélectionné depuis son parent³⁹

Insérer un nœud

```
function insertNode() {
  var newText = document.createTextNode("New Text");
  var newGraf = document.createElement("p");
  newGraf.appendChild(newText);

  var divMod = document.getElementsByTagName("div")[0];
  var allGrafs = divMod.getElementsByTagName("p");
  var oldGraf = allGrafs.item(0); // position

  divMod.insertBefore(newGraf, oldGraf);
}
```

³⁹. <https://developer.mozilla.org/en-US/docs/Web/API/Node/parentNode>

- Création du nouveau nœud :
 - `allGrafs` contient tous les éléments `p`
 - `lastGraf` contient le dernier du tableau `allGrafs`
- Insertion :
 - Recherche du parent
 - Recherche du frère gauche
 - Insertion depuis le parent

Avec jQuery

- Création et ajout :

```
var noeud = $('<p>Nouveau texte</p>'); // create node
$("body").append(noeud); // après le dernier fils
```

- Sélection et Suppression :

```
var noeud = $("p"); // select node(s)
noeud.remove();
```

Références

- Une réintroduction à JavaScript ⁴⁰
- [Things to avoid in JS][https://waspdev.com/articles/2025-06-13/things-to-avoid-in-javascript#not_using_modules]
- How does it feel to learn JS in 2016 ⁴¹
- Référence MDN ⁴²
- Tutoriels The Modern JS Tuto ⁴³ w3schools ⁴⁴ LearnJS ⁴⁵
- Outils de développement Chrome et Firefox (F12, Ctrl+Shift I)
- Visualisation du DOM ⁴⁶
- Outils web
 - JSFiddle ⁴⁷
 - JSLint ⁴⁸

40. https://developer.mozilla.org/fr/docs/Web/JavaScript/Une_r%C3%A9introduction_%C3%A0_JavaScript

41. <https://hackernoon.com/how-it-feels-to-learn-javascript-in-2016-d3a717dd577f>

42. <https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference>

43. <https://javascript.info/>

44. <http://www.w3schools.com/js/>

45. <https://learnjavascript.online/>

46. <http://bioub.github.io/dom-visualizer/>

47. <https://jsfiddle.net/>

48. <http://www.jslint.com/>

Sources