

05.JavaScript & DOM

23 janvier 2023

Développement web il3

JavaScript & DOM

HE-Arc (DGR) 2022

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
- One Page Apps
- Implémentations hors-browser
 - Node.js, Spidermonkey, Rhino
 - script d'app (Qt, Notepad++, ...)

- Langage cible de compilateurs : emscripten¹, WebAssembly²
- Embarqué : Espruino³, robotique : Node Bots⁴, CylonJS⁵
- Applications Desktop : Electron^{6, 7}

*Script

- ECMAScript : Norme depuis 1997
 - Juin 2022 : ECMA-262 13th edition⁸
 - Support⁹ des différentes implémentations
 - Conversions avec BabelJS¹⁰
- JavaScript : implémentation Firefox (réf. MDN)
- Variantes (à transpiler) :
 - Typescript¹¹ : variante fortement typée, avec des classes (MS)
 - Coffescript¹²
 - * sucre syntaxique
 - * compilé -> js

JavaScript

- Différentes implémentations¹³ : navigateur, srv, apps, ...
- Permissif : du mauvais code est peu maintenable
 - Design Patterns¹⁴
 - Bonnes pratiques¹⁵
- Interface pour scripter le navigateur
 - Accès et modification du contenu via DOM

¹<https://github.com/kripken/emscripten/wiki>

²<http://webassembly.org/>

³<http://www.espruino.com/>

⁴<https://nodebots.io/>

⁵<https://cylonjs.com/>

⁶<https://electronjs.org/>

⁷<https://sciter.com/>

⁸<https://www.ecma-international.org/publications-and-standards/standards/ecma-262/>

⁹<http://kangax.github.io/compat-table/es2016plus/>

¹⁰<https://babeljs.io/>

¹¹<https://www.typescriptlang.org/>

¹²<http://coffeescript.org/>

¹³https://en.wikipedia.org/wiki/List_of_ECMA_Script_engines

¹⁴<https://addyosmani.com/resources/essentialjsdesignpatterns/book/>

¹⁵<http://jstherightway.org/>

- Bookmarklets¹⁶, exemples¹⁷
- 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
 - Prototypes¹⁸
 - Fermetures¹⁹
 - Promesses²⁰ (MDN²¹, Google²²)

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 :

¹⁶<http://www.howtogeek.com/125846/the-most-useful-bookmarklets-to-enhance-your-browsing-experience/>

¹⁷<http://www.hongkiat.com/blog/100-useful-bookmarklets-for-better-productivity-ultimate-list/>

¹⁸https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Le_mod%C3%A8le_objet_JavaScript_en_d%C3%A9tails

¹⁹http://www.w3schools.com/js/js_function_closures.asp

²⁰<https://www.promisejs.org/>

²¹https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Promise

²²<https://developers.google.com/web/fundamentals/getting-started/primers/promises>

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

JavaScript dans la page web

- Éléments `<script>` exécutés dans l'ordre de la page
- Conseillé de les placer en fin de page²³
- Événements (`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`²⁴

²³https://developer.yahoo.com/performance/rules.html#js_bottom=

²⁴<https://www.w3.org/TR/html5/scripting-1.html#the-script-element>

Unobstrusive JS²⁵

- 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
 - 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 : DOMContentLoaded²⁶

Node.js²⁷ / Deno²⁸

- Node.js : une implémentation hors navigateur
 - environnement d'exécution + bibliothèques
 - event driven, non-blocking IO -> scalable
 - V8 engine
 - scripts exécutables sans navigateur
 - npm²⁹ : gestionnaire de paquets
 - gulp : make js
- Exemples³⁰ d'applications

²⁵https://en.wikipedia.org/wiki/Unobtrusive_JavaScript

²⁶<https://developer.mozilla.org/en/docs/Web/Events/DOMContentLoaded>

²⁷<https://nodejs.org>

²⁸https://www.reddit.com/r/node/comments/nx9qqr/deno_vs_nodejs_a_comparison_you_need_to_know/

²⁹<https://www.npmjs.com>

³⁰<https://colorlib.com/wp/npm-packages-node-js/>

- gulp, grunt, bower, yarn
- browserify
- serveur http
- express, cordova, forever, dev, pm2, karma, sails, phantomjs
- Tuto³¹, Playground³²

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

DOM

```
<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

³¹<https://www.tutorialspoint.com/nodejs/index.htm>

³²<https://runkit.com>



FIG. 1 : DOM tree

```
querySelector(), querySelectorAll(),
getElementById(), getElementsByTagName(), getElementsByClass(),
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 :
 - 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 allGraf = document.getElementsByTagName("p");

    if (allGraf.length > 1) {
        var lastGraf = allGraf.item (allGraf.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 dernier du tableau allGrafS
- Suppression :
 - Suppression du nœud sélectionné depuis son parent³³

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);
}

```

- 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 :

³³<https://developer.mozilla.org/en-US/docs/Web/API/Node/parentNode>

```
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 JavaScript³⁴
- How does it feel to learn JS in 2016³⁵
- Référence MDN³⁶
- Tutoriels The Modern JS Tuto³⁷ w3schools³⁸
- Outils de développement Chrome et Firefox (F12, Ctrl+Shift I)
- Visualisation du DOM³⁹
- Outils web
 - JSFiddle⁴⁰
 - JSLint⁴¹

Sources

³⁴https://developer.mozilla.org/fr/docs/Web/JavaScript/Une_r%C3%A9introduction_%C3%A0_JavaScript

³⁵<https://hackernoon.com/how-it-feels-to-learn-javascript-in-2016-d3a717dd577f>

³⁶<https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference>

³⁷<https://javascript.info/>

³⁸<http://www.w3schools.com/js/>

³⁹<http://bioub.github.io/dom-visualizer/>

⁴⁰<https://jsfiddle.net/>

⁴¹<http://www.jshint.com/>