

JAVASCRIPT

Cr   par [Alexandre Rivet](#)

INTRODUCTION

DÉFINITION

- Créé en 1995
- Un langage de programmation les plus populaires et fait partie des *standards* avec le HTML et CSS
- Langage interprété par les principaux navigateurs web: Google Chrome, Safari, Firefox, Edge, etc.
- Son évolution est gérée par le group ECMA international

FONDAMENTAUX

- JavaScript est un langage dynamique
- JavaScript est un langage (principalement) côté client
- JavaScript est un langage interprété
- JavaScript est un langage orienté objet

LANGAGE DYNAMIQUE

- Mettre à jour le contenu des pages web sans devoir changer le code manuellement
- En opposition aux langages *statiques* tels que HTML et CSS. Pour rappel, le HTML est un langage de balisage utilisé pour structurer le contenu d'une page web et le CSS lui sera là pour mettre en forme et donc styliser ce contenu
- Intervient donc pour manipuler le contenu HTML ou les styles CSS

LANGAGE (PRINCIPALEMENT) CÔTÉ CLIENT (1/2)

- Un site web est un ensemble de ressources et de fichiers liés entre eux
- Pour accéder à un site Web de n'importe où, il faut l'héberger sur un serveur
- Un serveur est un ordinateur connecté à d'autres serveurs qui va héberger les fichiers et les envoyer à un client sur sa demande

LANGAGE (PRINCIPALEMENT) CÔTÉ CLIENT (2/2)

- Lorsqu'on accède à un site web via un navigateur, le serveur envoie donc tous les fichiers puis le navigateur se charge de les **exécuter côté client**
- En opposition, un langage côté serveur est exécuté côté serveur et les navigateurs ne sont pas capables de les comprendre

Note: Le JavaScript est principalement côté client mais on pourrait aussi s'en servir côté serveur à condition de l'utiliser dans un environnement adéquat (avec Node.js notamment)

LANGAGE INTERPRÉTÉ

- Langage interprété: Capable d'être exécuté directement sous condition d'avoir le bon **interpréteur** (ce que possède chaque navigateur web)
- En opposition au langage **compilé** tel que le langage C qui transforme d'abord en une autre forme pour pouvoir exécuter le code

LANGAGE ORIENTÉ OBJET

Détaillé ici

ENVIRONNEMENT DE TRAVAIL

EDITEURS

- Editeurs de textes simples: [Sublime Text](#), [Visual Studio Code](#) (recommandé)
- Des environnements de développement intégrés (IDE): [WebStorm](#), [IntelliJ IDEA](#)
- Editeurs en ligne: [CodePen](#), [JSFiddle](#)

NAVIGATEURS

- Google Chrome (recommandé)
- Mozilla Firefox
- Safari
- Microsoft Edge

DEBOGAGE

PREMIER PROJET

LA PAGE HTML

```
<!doctype html>
<html>
  <head>
    <title>Hello World Page</title>
  </head>
  <body>
    Hello World!
  </body>
</html>
```

ECRIRE DU JAVASCRIPT - DANS UNE BALISE HTML

Résultat	index.html
<pre><!doctype html> <html> <head> <title>Hello World Page</title> </head> <body> <button onclick="alert('Hello World!')">Hello</button> </body> </html></pre>	

De nouvelles techniques permettent d'éviter ce genre de syntaxe qui est même déconseillé et considéré comme une mauvaise pratique.

Aujourd'hui le web et donc les composants sont de plus en plus complexes et donc la séparation des morceaux de code est la norme aujourd'hui.

ECRIRE DU JAVASCRIPT - DANS LA PAGE HTML

Résultat	index.html
<pre><!doctype html> <html> <head> <title>Hello World Page</title> </head> <body> <button>Hello</button> <script type="text/javascript"> document.querySelector('button').onclick = function() { alert('Hello World!'); } </script> </body> </html></pre>	

Cette solution est meilleure que la précédente mais encore une fois pas celle à retenir pour les mêmes raisons que la précédente. Cela est surtout vrai si on écrit du code que l'on souhaite réutiliser entre plusieurs pages

ECRIRE DU JAVASCRIPT - DANS UN FICHIER SÉPARÉ

Résultat	index.html	main.js
	<pre><!doctype html> <html> <head> <title>Hello World Page</title> </head> <body> <button>Hello</button> <script src="./main.js" type="text/javascript"></script> </body> </html></pre>	

Cette solution sera donc la meilleure des 3 vues qui permettra une meilleure réutilisation et une meilleure scalabilité du code. Si on change le code à un seul endroit, la modification se répercutera partout.

ORDRE D'EXÉCUTION DU CODE PAR LA NAVIGATEUR (1/3)

Par défaut un navigateur va lire et exécuter le code dans l'ordre de son écriture. Il va donc analyser chaque ligne et en fonction faire des choses additionnels. Par exemple, quand le navigateur va croise une balise script, il va s'arrêter dessus (donc bloquer le reste du traitement HTML), pour charger le code JavaScript et exécuté.

ORDRE D'EXÉCUTION DU CODE PAR LA NAVIGATEUR (2/3)

Admettons que le codé soit exécuter avant que le reste de la page et que le code fasse référence à des éléments du DOM, cela posera un problème car les éléments ne seront pas présent dans la page et donc le code plantera.

ORDRE D'EXÉCUTION DU CODE PAR LA NAVIGATEUR (3/3)

Il existe 3 manières de contourner ça:

- Mettre tout le JavaScript que l'on souhaite exécuter dans

```
document.addEventListener('DOMContentLoaded',  
function() {})
```

- Mettre ses imports de JavaScript en fin de `body` pour qu'il soit chargé et exécuté en dernier
- Ajouter un attribut `async` ou `defer` sur la balise `script` qui permettra de mieux ordonner le chargement ou même de charger en asynchrone

SYNTAXE & VARIABLES

STRUCTURES DE CONTRÔLE

FONCTIONS

OBJETS ET TABLEAUX

MANIPULATION DU DOM

GESTION DES ERREURS

PROGRAMMATION ORIENTÉE OBJET

ASYNCHRONICITÉ ET REQUÊTES AJAX

MODULES & OUTIL DE DÉVELOPPEMENT