



Javascript ES6

Louis Harang - Dylan Correia
Wizards Technologies



Qui sommes-nous ?



Dylan Correia

Formation :

- DUT Informatique (Orsay)
- Master Ingénierie du Web (ESGI)

24 ans

Email : dcorreia@wedigital.garden



Louis Harang

Formation :

- Master Ingénierie du Web (ESGI)

27 ans

Email : cours@narah.io

Développeurs Full Stack @ Wizards Technologies



Faisons connaissance

Présentez vous !

- Prénom
- Dans quelle boîte vous travaillez et avec quelles technos
- Ce que vous connaissez du javascript
- Ce que vous attendez du cours

Sommaire

Introduction

- Histoire
- ECMAScript
- Utilisation du Javascript
- C'est quoi le DOM ?
- Manipuler le DOM *

Les bases de Javascript ES6

- Let, const (& var)
- Fonctions *
- Les events *
- Itérations & Conditions *
- Manipuler les strings *

Manipuler des tableaux et des objets

- Les tableaux et les objets
- Les méthodes des tableaux *
- Les méthodes des objets *

Programmation fonctionnelle

- Les closures
- Les callbacks
- Les IIFE *

Javascript avancé :

- Promesses *
- Modules ES *
- Fonctions fléchées
- Retour implicite
- Object literal *
- Paramètres par défaut
- Déstructuration, spread operator & rest operator *

Les classes

- Introduction
- Créer sa première classe *
- Prototype
- Héritage *

Aller plus loin

- Les dernières versions de ECMA
- Les frameworks Front
- Typescript





Modalités d'évaluation

50% TP notés + 50% projet de groupe



Organisation des TP

- Chaque TP sera **numéroté**
- Le rendu doit être mis sur **Github** et envoyé par email
- Chaque repository doit comporter un readme avec :
 - Votre nom
 - Votre prénom
 - Votre adresse email
- Un seul repo pour tous les TP avec 1 dossier par TP comme ci-dessous:
 - TP1
 - TP2
 - TP3
 - ...
 - README.MD
- 1 fichier par exercice sauf si mention contraire dans l'énoncé
- Les TP sont des travaux personnels, si il y a la moindre ressemblance dans le code entre plusieurs personnes = 0 pour tout le monde



Projet de groupe

Plus loin dans le cours nous vous présenterons le sujet pour le projet de Javascript.

- Le sujet sera imposé
- Travail en groupe de 1 à 4 personnes (max).
- Le projet devra contenir un README avec le nom + prénom de tous les développeurs ainsi que vos emails.
- Vous devez utiliser **github** obligatoirement
- Plagiat ou triche = 0
- Tous les membres du groupe doivent travailler de manière égale sur le projet (nous regarderons les commits)

Le barème du projet vous sera transmis en même temps que le sujet.



Introduction



Histoire

Origine du javascript

Développé en dix jours en **mai 1995** pour le compte de la **Netscape Communications Corporation** par **Brendan Eich**. Le langage est inspiré de plusieurs langages, notamment **Java** mais en simplifiant la syntaxe.

Ce nouveau langage est appelé **LiveScript**.

Quelques jours avant sa sortie, Netscape change le nom de **LiveScript** pour **JavaScript**.

En **mars 1996**, Netscape met en œuvre le moteur JavaScript dans son navigateur web. Le succès de ce navigateur contribue à l'adoption rapide de JavaScript dans le **développement web orienté client**.

Netscape soumet alors **JavaScript** à **ECMA International** pour standardisation.

ECMAScript

ECMA quoi ?! 1/2

WIZARDS TECHNOLOGIES



Ecma International - *European association for standardizing information and communication systems* - est une organisation de standardisation active dans le domaine de l'informatique.

ECMA développe des standards sur les langages de script et de programmation, les technologies de télécommunications les produits de sécurité et pleins d'autres sujets informatiques.

Les travaux de standardisation débutent en **novembre 1996** et se terminent en **juin 1997**. Ces travaux donnent naissance au standard ECMA-262 qui spécifie le langage **ECMAScript**.



ECMAScript

Le versionning de ECMAScript 2/2

- **ECMAScript 1** : Juin 1997
- **ECMAScript 2** : Juin 1998
- **ECMAScript 3** : Décembre 1999
- **ECMAScript 4** : Abandonné car trop complexe, les features de la version 4 ont été réintégrées dans la version 6
- **ECMAScript 5** : Décembre 2009, pleins d'ajouts importants
- **ECMAScript 6** : Juin 2015, révolution de la syntaxe et des fonctionnalités
- **ECMAScript 7 à ?** : Depuis la version 6, les versions ECMAScript prendront le nom d'ES<Année en cours>. Chaque année une nouvelle version sort avec des nouveautés.

Aujourd'hui nous sommes donc à **ES 2021**

Dans ce cours nous parlons d'ES 6 (ES 2015), ce n'est pas que nous allons nous arrêter à la version 6, mais simplement ES6 est le nom officieux du **Javascript moderne post-ES6**.



Utilisation du Javascript

Où et comment

Où peut on exécuter du javascript ?

Historiquement, dans votre **navigateur**

Aujourd'hui avec Node, il est aussi possible d'exécuter du **javascript dans le terminal**.

Il est aussi possible de lancer des **scripts JS dans MongoDB**.

Et même dans des **applications desktop** (Electron) et **mobile** (Flutter, React Native...) !

Dans notre cours, nous allons nous concentrer sur l'exécution du javascript dans le **navigateur** et dans le **terminal**.



Utilisation du Javascript

Pour commencer, vous pouvez exécuter du javascript dans la console de votre navigateur ou alors créer un page HTML avec une balise `<script>`

```
<!DOCTYPE html>
<html>
<head>
  <title>My first page</title>
</head>
<body>
</body>
<script type="text/javascript">
  alert('hello world !');
</script>
</html>
```

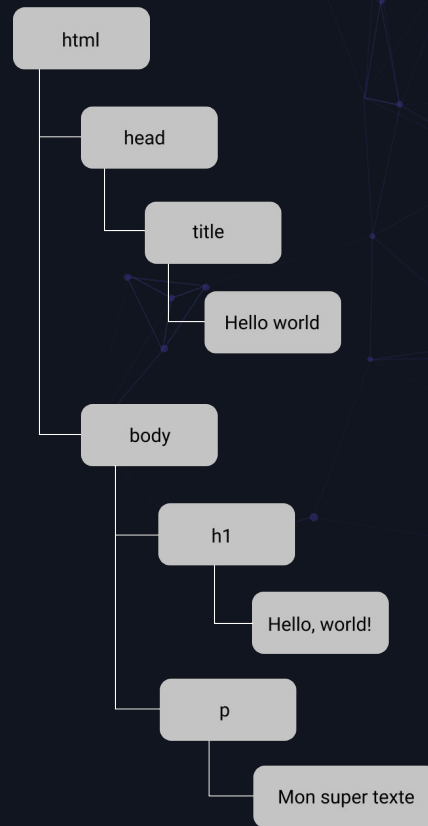


C'est quoi le DOM ?

L'arborescence d'une page web

Le **DOM** (Document Object Model) est une représentation d'un document HTML sous forme d'un **arbre d'objets**.

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <title></title>
  </head>
  <body>
    <h1> Hello, world !</h1>
    <p>Mon super texte</p>
  </body>
</html>
```





Manipuler le DOM

Browser Object Model (BOM)

Le BOM est le coeur du Javascript dans le navigateur. Le BOM est composé de plusieurs objets qui permettent d'ajouter des fonctionnalités et de l'interaction avec une page web **indépendamment de son contenu**. Attention, même si les BOM sont semblables entre les navigateurs, ils ne sont pas tous forcément disponible. Chaque navigateur en a un implémentation différente.

- **window** : supporté par tous les navigateurs, **window** représentent la fenêtre du navigateur.
- **window.location** : permet de manipuler l'URL de la page
- **window.history** : permet de gérer l'historique de navigation de la page
- **window.document (document)** : document, c'est la représentation de la page web, c'est le DOM
- **window.alert (alert)** : permet d'afficher des modales d'alertes



Manipuler le DOM

Récupérer des infos

Via l'objet **document** on a accès à plusieurs méthode permettant de retrouver des éléments du DOM.

```
/*
 * On récupère l'element avec l'id header
 * Si il y a plusieurs element avec header,
 * la méthode retournera la première occurrence
 */
const element = document.getElementById('header');

/*
 * On récupère les elements avec la classe button
 * La méthode retourne un tableau d'element
 */
const elements = document.getElementsByClassName('button');

/*
 * On récupère les elements avec le tag li
 * La méthode retourne un tableau d'element
 */
const elements = document.getElementsByTagName('li');
```




Manipuler le DOM

Récupérer des infos

Via l'objet **document** on a accès à plusieurs méthode permettant de retrouver des éléments du DOM.

```
/*
 * On peut récupérer des informations sur un element du DOM
 * Ici on récupère la valeur de l'attribut src
 */
const element = document.getElementById("avatar").getAttribute("src");

/*
 * On peut aussi récupérer d'autres champs
 */
const element = document.getElementById("avatar").value("src");

/*
 * Pour modifier le texte entre la balise ouvrante et fermante d'un element du DOM
 * on utilise innerHTML
 */
const element = document.getElementById("bio").innerHTML = "Ma super bio";

/*
 * On peut aussi modifier le CSS d'un element
 */
const element = document.getElementById("title").style.color = "red";
```



Manipuler le DOM

Créer des éléments HTML

On peut créer des éléments dans le DOM directement en javascript, pour cela il faut tout d'abord créer un élément puis l'attacher à un autre élément du DOM.

```
// On créer un élément p avec la méthode createElement
const myElement = document.createElement("p");
// On personnalise notre élément
myElement.innerHTML = "Hello world !";
myElement.style.color = "red";

// Il faut attacher notre nouvel élément à un élément existant dans le DOM
// On va récupérer le body et lui attacher notre élément
document.getElementsByTagName('body')[0].appendChild(myElement);
```

Manipuler le DOM

Créer des éléments HTML

WIZARDS TECHNOLOGIES



On peut bien aussi supprimer un élément du DOM ou encore remplacer un élément par un autre

```
const element = document.getElementById('title');  
  
// On supprime avec removeChild  
document.removeChild(element);  
  
// On peut aussi remplacer un élément avec replaceChild  
document.replaceChild(newElement, oldElement);
```



Manipuler le DOM

Créer des éléments HTML

On peut aussi utiliser **document.write** pour écrire directement dans le stream HTML. **Attention** cette méthode remplace tout le HTML de la page.

```
document.write("Hello world !");
```



Méthodes à savoir

Pour vous aider à développer et déboguer vos applications client javascript, vous pouvez utiliser les méthodes ci-dessous pour dump vos variables.

```
// console.log permet d'afficher des variables dans la console du navigateur  
console.log("foo"); // affiche dans la console : "bar"  
  
// alert permet d'afficher une pop up dans votre navigateur  
alert('hello world');
```



TP 1

Manipulation du DOM

Pour chaque exercice, faites l'exo dans un fichier HTML avec une balise script

1. Coder un script JS qui change la couleur de fond du body
2. Créer une page HTML avec 3 balises "p" contenant du texte, coder un script permettant de mettre en gras le 1er élément, en italique le 2ème et en uppercase le 3ème
3. Créer une page HTML avec 3 images, coder un script qui remplace l'attribut src de la 1ère et 2ème image par celui de la 3ème
4. Coder un script qui permet d'ajouter un élément div avec un fond violet, une largeur de 100% et une hauteur de 300px. Créer un autre élément "h1" avec une couleur de texte blanche, en majuscule. Attacher l'élément H1 à l'élément div.
5. Récupérer une page HTML sur internet, puis rajouter un script permettant de supprimer tous les éléments "p"
6. Créer une page HTML avec une liste "Toto" "Tata" "Titi" "Tutu". Créer un nouvel élément li et remplacer le 2ème élément de la liste par le nouvel élément.
7. Récupérer une page HTML sur internet, avec **console.log** afficher la largeur et la couleur de fond de la balise body