

# 420-1WE-BB

## Web dynamique

JavaScript - 1

# Objectif pédagogique

- ▶ Comprendre l'origine de JavaScript
- ▶ Visualiser l'engin (ou le moteur) d'exécution de JavaScript
- ▶ Comprendre les avantages et les limites de JS

# Qu'est ce que JavaScript

- ▶ JavaScript a été créé afin de dynamiser les pages HTML.
- ▶ Les programmes de ce langage sont appelés scripts et peuvent être écrits directement dans la page HTML et lancer au chargement de cette dernière.
- ▶ Java  $\neq$  JavaScript
  - ▶ Le langage est interprété et ne requiert aucune compilation.
  - ▶ Il est un peu similaire dans sa syntaxe au langage Java, mais il diffère au niveau de sa structure et exécution.

# La genèse de JavaScript

- ▶ Inventé par Brendan Eich, de Netscape, il a nommé le langage Mocha puis LiveScript, mais Java connaissait un essor important, alors il a été renommé JavaScript (un coup marketing), a été lancé en 1995.
- ▶ Aujourd'hui JavaScript est un langage indépendant standardisé par ECMA International d'où son nom réel peu sexy de ECMAScript.

# Les versions

Édition	Publié en	Nom	Changements importants
5.1	June 2011		This edition 5.1 of the ECMAScript
6	June 2015	ECMAScript 2015 (ES2015)	Nouvelle syntaxe pour écrire des applications complexes, incluant les declarations de classes
7	June 2016	ECMAScript 2016 (ES2016)	Déclaration de Variables avec portée de bloc (block-scoping) let & const. programmation asynchrone (async/await)
8	June 2017	ECMAScript 2017 (ES2017)	Promesses
9	June 2018	ECMAScript 2018 (ES2018)	
10	June 2019	ECMAScript 2019 (ES2019)	

# L'engin d'exécution

- ▶ Chaque navigateur possède un moteur JavaScript qui lui permet d'exécuter le code JS.
- ▶ Ce moteur peut aussi être utilisé à l'extérieur du navigateur et est très utilisé en arrière plan (ex.: node.js)

Pourcentage de support pour les versions de moteur JS

Moteur JS	Navigateur	ES5	ES6	ES7	2016+
Chakra	Edge <= v18	100%	96%	100%	39%
SpiderMonkey	FireFox > v75	100%	98%	100%	87%
V8	Chrome > v81	100%	98%	100%	100%
JSCore	Safari 13.1	99%	99%	100%	88%

# Le fonctionnement du moteur JS



# Que peut faire JavaScript dans le navigateur

- ▶ Ajouter, modifier du code HTML de la page et/ou le contenu existant, modifier les styles. Document Object Model (DOM)
- ▶ Réagir aux actions de l'utilisateur,
  - ▶ clics de souris,
  - ▶ mouvements de pointeur
  - ▶ appuis sur des touches.
- ▶ Envoyer des requêtes sur le réseau à des serveurs distants, télécharger et envoyer des fichiers (technologies REST, AJAX et COMET).
- ▶ Obtenir et définir des cookies, poser des questions au visiteur, afficher des messages.
- ▶ Se souvenir des données du côté client (“stockage local”).

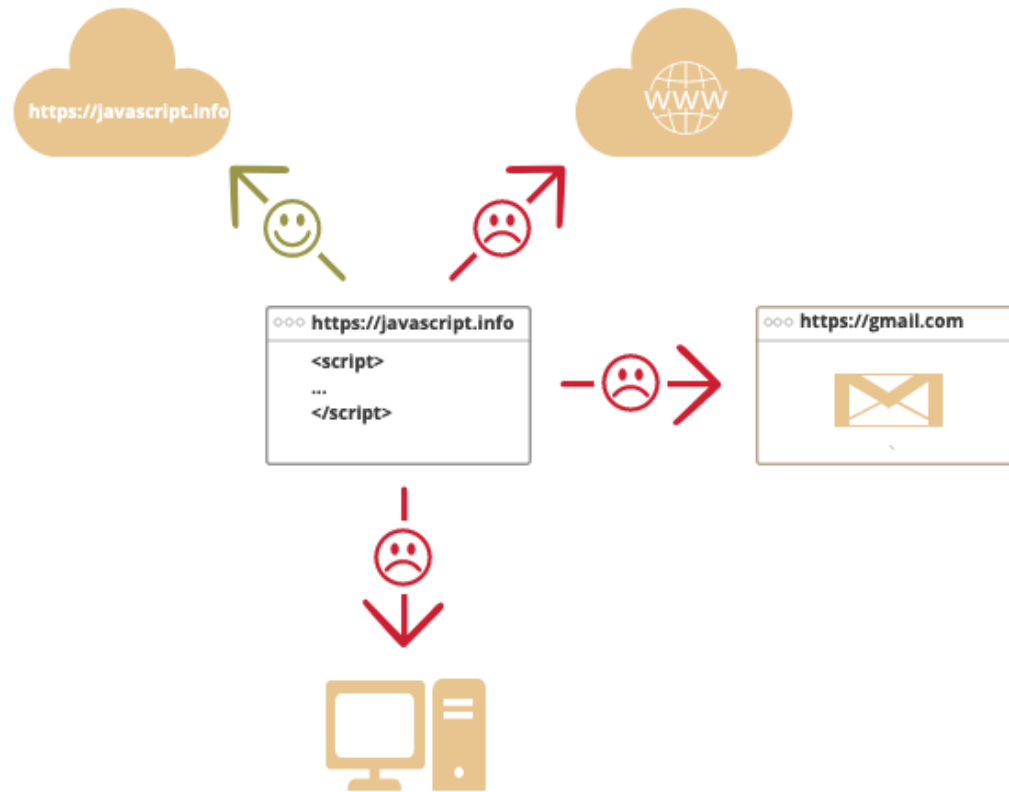


# Qu'est-ce que JavaScript NE peut PAS faire?

- ▶ Accéder au système d'exploitation
- ▶ Accéder au système de fichiers
- ▶ Accéder à d'autres fenêtres, sauf celles de même origine « same origin policy ».
- ▶ Accéder à d'autres domaines que celui courant, à moins d'entente CORS.

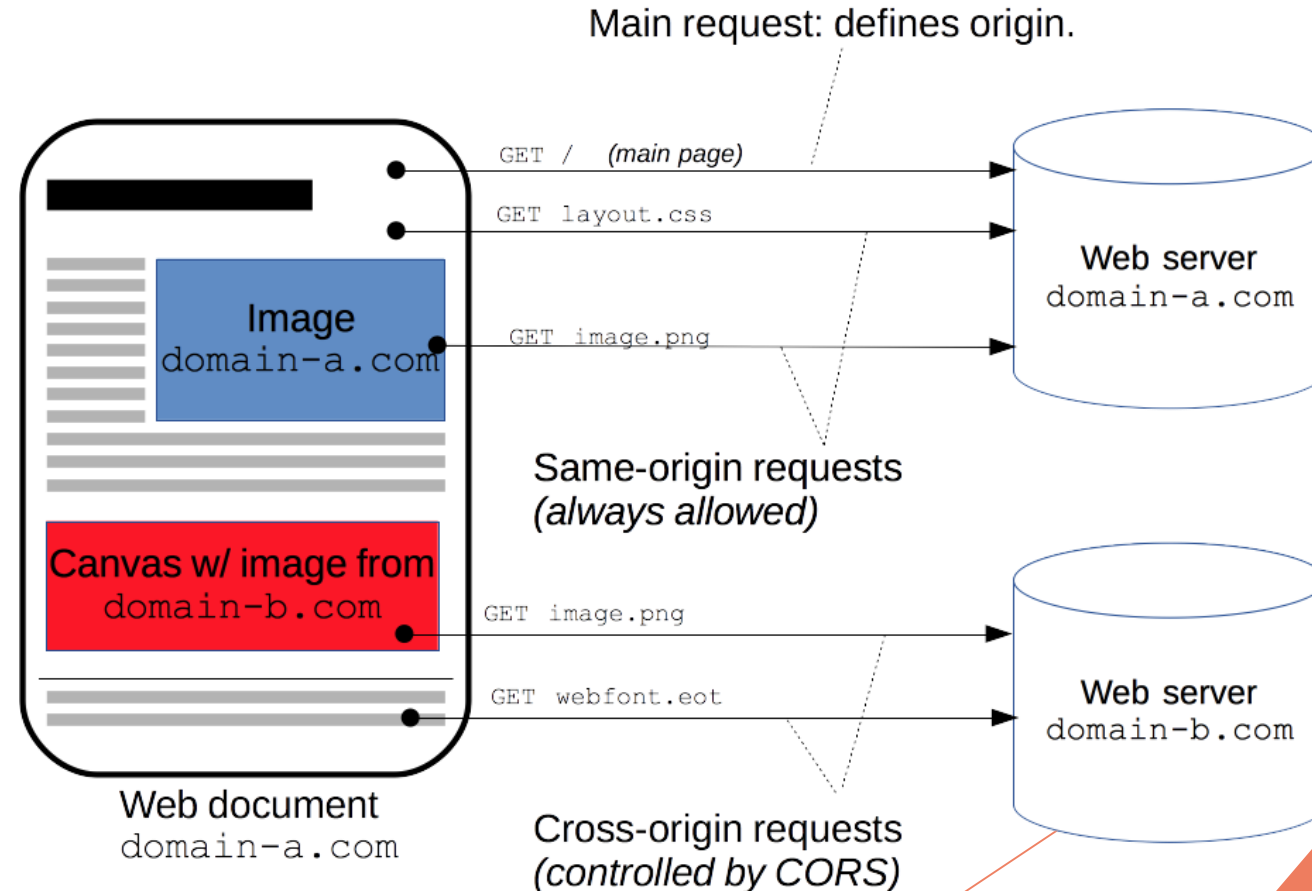
# Que peut faire JS?

- En tant que langage sûr il ne fournit pas d'accès de bas niveau



# Cross-Origin Resource Sharing

- Traduction libre: Politique de partage de ressources d'origine croisé




# Avantages de JS dans le navigateur

- ▶ Intégration complète avec HTML / CSS.
- ▶ Les choses simples sont faites simplement.
- ▶ Pris en charge par tous les principaux navigateurs et activé par défaut.

# Langages au dessus de JS

- ▶ Il existe des langages qui offrent des fonctionnalités additionnelles pour le développement et qui par la suite s'exécute à l'aide de JavaScript grâce à un processus de transpilation.
  - ▶ TypeScript se concentre sur l'ajout de "typage strict des données" pour simplifier le développement et la prise en charge de systèmes complexes. Il est développé par Microsoft.
  - ▶ Flow ajoute également la saisie de données, mais de manière différente. Développé par Facebook.

# Votre premier programme JS

Users > jifbrodeur > Dropbox > BdB > Cours > 420-2CW-BB > Cours > Cours 9 - JS >  bonjour.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="UTF-8" />
5      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6      <title>Document</title>
7    </head>
8    <body>
9      <p>Avant le script...</p>
10
11     <script>
12       | alert("Bonjour le monde");
13     </script>
14
15     <p>...Après le script.</p>
16   </body>
17 </html>
18
```

# La balise <script>

- ▶ Comme indiqué précédemment est en ligne (in-line)
- ▶ Script externe
- ▶ Le fichier de script est attaché à HTML avec l'attribut src :
  - ▶ `<script src="/chemin/vers/script.js"></script>`
  - ▶ `<script src="https://cdnjs.cloudflare.com/ajax/libs/lodash.js/3.2.0/lodash.js"></script>`
- ▶ Le type de balise script ne peut être à la fois en-ligne et externe

# Exercice

- ▶ Faites une page html avec le code JS pour afficher l'alert suivante: « Je suis nom »
- ▶ Modifier la page précédente afin de la relier à un fichier JS externe intitulé alert.js et ce fichier JS doit contenir l'alerte à afficher



# Structure de code JS

- ▶ Dans un script JS il peut y avoir autant d'instructions dans le code que nous le souhaitons. Une autre instruction peut être séparée par un point-virgule.
- ▶ Par exemple, ici nous divisons le message en deux :

```
alert('Bonjour'); alert('Le monde');
```

- ▶ Chaque instruction est généralement écrite sur une ligne distincte - le code devient donc plus lisible :

```
alert('Hello');  
alert('World');
```

# Les points virgules

- ▶ Grand sujet de débat

# Les commentaires

- ▶ `//` Ce commentaire occupe une ligne à part

```
    alert('Hello');
```

```
    alert('World'); // Ce commentaire suit l'instruction
```

```
/* Un exemple avec deux messages.
```

```
C'est un commentaire multiligne.
```

```
*/
```

```
    alert('Hello');
```

```
    alert('World');
```

- ▶ Utiliser les raccourcis !
- ▶ Dans la plupart des éditeurs, une ligne de code peut être commentée par le raccourci `Ctrl+/` pour un commentaire sur une seule ligne et quelque chose comme `Ctrl+Shift+/`