**Une première définition du JavaScript**

**COURS JAVASCRIPT**

Le JavaScript est un langage de programmation créé en 1995. Le JavaScript est aujourd’hui l’un des langages de programmation les plus populaires et il fait partie des langages web dits « standards » avec le HTML et le CSS. Son évolution est gérée par le groupe ECMA International qui se charge de publier les standards de ce langage.

On dit que le HTML, le CSS et le JavaScript sont des standards du web car les principaux navigateurs web (Google Chrome, Safari, Firefox, etc.) savent tous « lire » (ou « comprendre » ou « interpréter ») ces langages et les interprètent généralement de la même façon ce qui signifie qu’un même code va généralement produire le même résultat dans chaque navigateur.

Pour définir ce qu’est le JavaScript et le situer par rapport aux autres langages, et donc pour comprendre les intérêts et usages du JavaScript il faut savoir que :

* Le JavaScript est un langage dynamique ;
* Le JavaScript est un langage (principalement) côté client ;
* Le JavaScript est un langage interprété ;
* Le JavaScript est un langage orienté objet.

Pas d’inquiétude, on va définir le plus simplement possible ce que ces qualificatifs signifient !

**Le JavaScript, un langage dynamique**

Le JavaScript est un langage dynamique, c’est-à-dire un langage qui va nous permettre de générer du contenu dynamique pour nos pages web.

Un contenu « dynamique » est un contenu qui va se mettre à jour dynamiquement, c’est-à-dire changer sans qu’on ait besoin de modifier le code manuellement mais plutôt en fonction de différents facteurs externes.

On oppose généralement les langages « dynamiques » aux langages « statiques » comme le HTML et le CSS. Illustrons les différences d’utilisation entre ces types de langage en discutant des possibilités du HTML, du CSS et du JavaScript.

Pour rappel, le HTML est un langage de balisage (langage qui utilise des balises) qui est utilisé pour structurer et donner du sens aux différents contenus d’une page. Le HTML nous permet de communiquer avec un navigateur en lui indiquant que tel contenu est un titre, tel contenu est un simple paragraphe, tel autre est une liste, une image, etc.

Le navigateur comprend les différentes balises HTML et va alors afficher notre page à nos visiteurs en tenant compte de celles-ci.

Le contenu HTML ne va jamais être affiché tel quel, brut, mais des règles de mises en forme vont lui être appliquées. Ces règles de styles vont être définies en CSS. Le CSS va ainsi nous permettre d’arranger les différents contenus HTML de la page en les positionnant les uns par rapport aux autres, en modifiant la couleur des textes, la couleur de fond des éléments HTML, etc.

Le CSS va ainsi se charger de l’aspect visuel de notre page tandis que le HTML se charge de la structure (définir les contenus) de celle-ci.

Le HTML et le CSS forment ainsi un premier couple très puissant. Cependant, nous allons être limités si nous n’utilisons que ces deux langages tout simplement car ce sont des langages qui ne permettent que de créer des pages « statiques ».

Une page statique est une page dont le contenu est le même pour tout le monde, à tout moment. En effet ni le HTML ni le CSS ne nous permettent de créer des contenus qui vont se mettre à jour par eux-mêmes. Le CSS, avec les animations, nous permet de créer des styles pseudo-dynamiques mais tout de même prédéfinis.

C’est là où le JavaScript entre en jeu : ce langage va nous permettre de manipuler des contenus HTML ou des styles CSS et de les modifier en fonction de divers évènements ou variables. Un évènement peut être par exemple un clic d’un utilisateur à un certain endroit de la page tandis qu’une variable peut être l’heure de la journée

## JavaScript, API, librairies et framework

Le JavaScript en tant que langage correspond à un ensemble de structures de codes ou un ensemble d’éléments qu’on va pouvoir utiliser pour implémenter des fonctionnalités sur nos pages web.

Les API et les librairies JavaScript sont construites à partir de ces éléments de base du JavaScript et vont nous permettre d’utiliser des structures plus complexes déjà prêtes à l’emploi qui vont in-fine nous permettre de réaliser simplement des opérations qu’il aurait été très difficile de réaliser si on avait dû les coder entièrement à la main.

Une API (« Application Programming Interface » ou « Interface de Programmation ») est une interface qui nous permet d’utiliser facilement une application. Une application est un programme, c’est-à-dire un ensemble cohérent de code qui permet de réaliser certaines actions.

On utilise les API pour demander au programme d’effectuer certaines actions pour nous, comme par exemple afficher une carte d’une certaine ville à une certaine échelle (Google Maps API) ou pour afficher la liste de nos derniers Tweets (Twitter API) ou encore pour manipuler le contenu HTML d’une page web (DOM API).

Pour utiliser une API et donc l’application correspondante, il faudra généralement demander au propriétaire de l’application une clef qui va nous permettre de nous identifier.

Une librairie ou « bibliothèque » JavaScript est un ensemble de fichiers de code JavaScript homogènes (= qui se concentrent sur un aspect particulier du langage) qu’on va devoir télécharger pour les utiliser. Ces fichiers de code contiennent des structures de code prêtes à l’emploi qu’on va pouvoir utiliser immédiatement pour gagner du temps en développement. Parmi les librairies les plus célèbres, on peut notamment citer jQuery.

Il convient donc de ne pas confondre API et librairies : une librairie est un ensemble de fichiers qu’on va télécharger et contient un ensemble de structures de codes prêtes à l’emploi. Nous allons pouvoir choisir celles qui nous intéressent pour les intégrer dans nos propres scripts et ainsi gagner du temps de développement. Une API, de l’autre côté, va nous permettre d’utiliser une application qu’on n’a pas le droit de manipuler directement.

Finalement, un framework ou « cadre de travail » est relativement similaire dans son but à une « super librairie ». Les framework vont également nous fournir un ensemble de codes tout prêts pour nous faire gagner du temps en développement. La grande différence entre un framework et une librairie réside dans l’inversion du contrôle : lorsqu’on télécharge une librairie, on peut l’utiliser comme on le souhaite en intégrant ses éléments à nos scripts tandis que pour utiliser un framework il faut respecter son cadre (ses règles). Les framework JavaScript les plus connus aujourd’hui sont Angular.js et React.js.

Dans le début de ce cours, nous n’utiliserons bien évidemment pas d’API ni de librairie et encore moins de framework. Cependant, il reste intéressant de déjà définir ces différents termes pour vous donner une première « vue d’ensemble » des outils JavaScript.