Cours 1 : introduction – le web, Javascript, Vue

1. le web

Le *World Wide Web*, abrégé en *web*, est un système hypertexte (un ensemble de documents reliés par des liens hypertextes) basé sur le réseau internet, et permettant d'accéder à des *pages* regroupant différentes informations. On date généralement sa création de 1990, et on l'attribue à Tim Berners Lee, qui l'a conçu en travaillant au CERN.

Le *web* permet l'accès à des *ressources* identifiées par des URLs (*Uniform Resource Locator*). Ces ressources sont, en très grandes parties, des pages HTML interprétées par des navigateurs web (pour les plus connus : Google Chrome, Mozilla Firefox, Edge ou Opera, mais il en existe une très grande variété).

2. Javascript et son écosystème

2.1. Généralités

Javascript est un langage de programmation développé initialement sous le nom de *LiveScript* par Brendan Eich en 1995. Il a été standardisé par la norme ECMAScript dès 1997. JavaScript est aujourd'hui considéré comme un des langages, sinon le langage le plus populaire au monde. Il a donné naissance à de nombreux *supersets* du langage qui étendent ses fonctionnalités (TypeScript, PureScript, Closure, etc.).

2.2. Le typage

Le langage est typé :

Dynamiquement : le type d'une variable est inféré en typant l'expression qui lui affecte une valeur. Le type n'est pas décidé à la création de la variable, au contraire de C ou de Java.

Faiblement : il est possible de faire certaines opérations entre deux types différents sans avoir à faire de conversion de type explicite. Il est ainsi possible de concaténer une chaîne de caractère et un nombre entier, ce dernier étant automatiquement converti en chaîne de caractères.

2.3. Paradigmes

Le langage permet également d'utiliser plusieurs paradigmes de programmation de façon concomitante :

Orienté objets : JavaScript permet l'utilisation de classes, d'objets, de méthodes et d'attributs. D'autres entités habituelles des langages de programmation orientés objets n'y sont néanmoins pas présents (en particulier les interfaces et les énumérations).

Impératif : Les instructions JavaScript « classiques » sont exécutées séquentiellement pour modifier l'état interne du programme. Cela recouvre les déclarations, affectations, conditions, boucles, séquences.

Fonctionnel : les fonctions peuvent être passés en paramètres d'autres fonctions ou méthodes, permettant de les combiner pour créer des fonctions pures complexes et réutilisables.

2.4. Ecosystème

JavaScript est exécuté dans deux contextes : dans le navigateur pour offrir une plus grande interactivité à l'utilisateur, et au sein d'un environnement d'exécution côté serveur. L'environnement le plus utilisé est **NodeJS**, mais on peut également citer **Deno** ou **Bun**.

Les dépendances au sein de JavaScript sont gérés par un gestionnaire de dépendances. Le plus connu (utilisé par défaut au sein de NodeJS) est **NPM**, mais il en existe d'autres, en particulier **PNPM** ou **Yarn.**

Il est possible d'utiliser certains outils dédiés pour pouvoir faire facilement cohabiter différentes versions de NodeJS. Dans les différentes séances de travaux pratiques, je vous conseille personnellement l'utilisation de **NVM** (*Node Version Manager*).

3. VueJS et la mécanique MVVM

3.1. Généralités sur VueJS

VueJS est une librairie de rendu graphique dans la lignée d'autres librairies et cadriciels (ou *frameworks*) tels que : jQuery, AngularJS ou React. Elle permet d'étendre le HTML avec de nouveaux attributs (appelés « directives »), qui sont soit incluses dans la librairie en elle-même, soit définies par l'utilisateur. Ces directives donnent accès à de nouvelles fonctionnalités (boucles, rendu conditionnel, etc.).

La librairie possède également d'autres fonctionnalités essentielles, tels que les composants (des éléments HTML personnalisés permettant la réutilisation de parties de code). Certaines autres fonctionnalités sont également fournies par des librairies tierces (maintenues par la même équipe ou non), tel que le routage fourni par vue-router.

3.2. Le MVVM

VueJS fait partie d'un ensemble de librairies de rendu graphique regroupés sous l'acronyme MVVM (« *Model-View, View-Model* ») dont la principale caractéristique est d'être capable de maintenir une cohérence entre une vue (l'HTML rendu dans le navigateur de l'utilisateur) et un modèle (une structure de données en Javascript). Quand l'un est modifié, l'autre est mis à jour pour refléter ces modifications.

Les mises à jour peuvent se faire dans les deux sens :

- **De la vue vers le modèle :** Une case à cocher est cochée dans la vue, le booléen qui y est lié dans le modèle passe alors de « *false* » à « *true* »
- **Du modèle vers la vue :** Un élément est ajouté à un tableau dans le modèle, la vue est alors mise à jour pour afficher ce nouvel élément dans une liste représentant ce tableau.