TP nº 6 : VueJS

Le but de ce TP est d'apprendre à utiliser VueJS, présenté comme « un framework évolutif pour construire des interfaces utilisateur » en JavaScript.
L'essentiel du framework sert à concevoir la « vue » de l'application web développée, sous la forme d'une *page unique* décrite par des *templates*. Cette page se met à jour de façon « réactive » en fonction des évolutions du modèle de l'application.

Dans ce TP, à titre d'exemple de programmation de vue réactive, vous programmerez un genre de moteur de blog ou micro-publications qui s'exécute à 100% dans le navigateur (pas de *back end*, donc pas non plus de possibilité de montrer ses publications à d'autres utilisateurs : c'est une application complètement locale !).

I) Travail préparatoire

1. Écrire une page web contenant un élément `<div id="app">` servant à « *monter* » votre application VueJS. Ajoutez-y du texte dans un paragraphe (élément `<p>` dans le `<div>`).

2. Faites en sorte que votre page charge la bibliothèque VueJS, ainsi qu'un fichier `app.js` dans le même répertoire, créé par vos soins.

Déclarez dans `app.js` une « vue » VueJS (composant principal) qui sera rendue dans l'élément dont l'`id` est `"app"`.

3. Ajoutez un champs booléen `visible` à sa section `data` et faites en sorte que le texte ne soit rendu que si ce booléen vaut `true` (vérifiez que ça marche en modifiant `app.visible` dans la console JavaScript et en regardant si la page change).

4. Ajoutez une *check box* telle que le paragraphe est affiché si et seulement si la *check box* est cochée (utiliser l'attribut `v-model="visible"`).

Vérifiez bien que ça marche.

5. Maitenant ajoutez une propriété `publications` dans les `data` du composant principal, qui soit un tableau de textes que vous prendrez au hasard (par exemple en ouvrant le lien "page au hasard" dans Wikipedia !).

Faites en sorte votre page affiche maintenant, dans le `<div id="app">` un paragraphe par élément du tableau `publications` (utilisez l'attribut `v-for`).

6. Ajoutez une zone d'entrée de texte et un bouton de telle sorte que quand on valide ce formulaire, une nouvelle publication s'ajoute au tableau `publications`. Vérifiez que par la même occasion un nouveau paragraphe, dont le contenu est le texte qui avait été entré, s'affiche bien dans la page.

Conseil : définissez une fonction dans la section `methods` du composant, afin d'éviter d'avoir à insérer du code compliqué dans l'attribut `v-on` du formulaire.

7. Ajoutez une propriété dans `data` dont la valeur représentera le nombre de secondes depuis la création du composant principal (mettez-le à jour grâce à un `setInterval` appelé depuis le *hook* `mounted` dans le composant principal).

Ajoutez un paragraphe affichant en permanence, de façon réactive, le temps écoulé depuis le chargement de la page.

II) Avec des composants personnalisés

8. Créez un composant réutilisable `publication`, de telle sorte que le *template* du composant principal n'utilise plus la balise `<p>` mais la balise personnalisée `<publication>` à chaque fois qu'une nouvelle publication doit être affichée.

Ce composant a deux propriétés (section `props`) : son texte, et sa date de publication.

Le *template* du composant `publication` doit afficher un paragraphe dont le contenu est de la forme `date : texte`.

9. Dans le composant `publication`, ajoutez une *check box* qui sert à *« liker »* cete publication (l'état de cette *check box* est relé à une propriété booléenne de la section `data` du composant).

Ajoutez aussi une *check box* « `N'afficher que les publications likées.` » dans le composant principal, elle aussi reliée à une propriété booléenne (du composant principal), de telle sorte que quand la *check box* est cochée, seules les publications « likées » soient affichées.

(Bien réfléchir si le choix d'afficher ou non, avec l'attribut `v-if`, se fait dans le composant principal ou dans le composant `publication`. Réfléchir aussi à la façon de remonter ou de redescendre les informations, le cas échéant.)

10. Faites en sorte qu'avant chaque publication, à la place de la date, on affiche de façon réactive le nombre de secondes écoulées depuis sa publication.

Chaque paragraphe ressemble alors plutôt à : `Il y a xxx secondes: bla bla bla.`

(Conseil : utiliser une propriété « calculée », c'est à dire une fonction définie dans la section `computed` du composant, pour calculer la différence de temps.)