Exercice Initiation à VueJS 1

# Template de votre vue

<!DOCTYPE html>

<html lang="fr">

<head>

<meta charset="utf-8" />

<title>Manipulation de vue</title>

<script src="vue.js"></script>

</head>

<body>

<div id="app"></div>

<script>

let vm = new Vue({

el: "#app",

data: {},

});

</script>

</body>

</html>

# Exercice 1

À partir du code ci-dessus, créer une première application avec des propriétés

titre

firstname

lastname

Qui contiendront un message de bienvenue votre nom et prénom et que vous afficherez dans votre page HTML

# Exercice 2

À partir du code ci-dessus, ajouter dans le modèle une propriété message, valant la chaîne de caractères "Salut!", et à afficher dans la vue à l'aide d'une directive, dans un paragraphe, puis en vous passant de la directive.

# Exercice 3

À partir du code ci-dessus, créez une vue-modèle dans laquelle une des données, notée d, est la date du jour. Affichez dans un paragraphe l'année en cours à partir de cette date.

# Exercice 4 - v-model

Déclarer un Vue avec une propriété température.

Donner une valeur à la température celle ci sera en Celsius.

Afficher la température dans la page.

Faire une checkbox qui lorsque qu’on la check convertit la température en Fahrenheit en utilisant la formule suivante (F = \* 1.8 + 32.0)

# Exercice 5 - v-model

À partir du code ci-dessus, créez l'interface suivante :

* Boutons radio:  
  Rouge  
  Bleu
* Cases à cocher:  
  Voulez-vous afficher le paragraphe?

Paragraphe à afficher

Faites en sorte que la couleur du paragraphe soit choisie *via* les boutons radio, et que la case à cocher contrôle son affichage ou sa disparition.

# Exercice 6 - v-bind

À partir du code ci-dessus, créez un paragraphe contenant un texte quelconque. Au chargement de la page, *via* Vue, faites-en sorte que son attribut title vaille "title initial". Au click sur ce paragraphe, le title devient "title modifié".

# Exercice 7 - v-if - v-show

À partir du code ci-dessus, créez un paragraphe contenant un texte quelconque. Au chargement de la page, *via* Vue, faites en sorte que son attribut title vaille "title initial". Au clic sur ce paragraphe, le title devient "title modifié".

# Exercice 8 - v-for

À partir du code ci-dessus, créez un tableau de todo dans vos propriétés que vous afficherez en utilisant la directive v-for

# Exercice Extra!

On pourra ajouter une checkbox pour chaque élément de notre liste dans l’exercice suivant pour rayer un élément afin de permettre à l’utilisateur de le considérer terminé

Exercice Initiation à VueJS 2

# Exercice 1 Filtre

Déclarer une date et ajouter un filtre pour l’afficher qui utilise le format suivant pour son affichage:

MM/DD/YYYY hh:mm

# Exercice 2 Événement

Soit un selectbox contenant les valeurs suivantes:

Phoenix

Wolverine

Black Widow

Thor

Daredevil

Deadpool

Spiderman

Faites une méthode qui lorsque l'événement sélectionné est enclenché, affiche l'élément sélectionné

# Exercice 3 Événement personnalisé

Créer un composant qui permet d’afficher dans un paragraphe la valeur que vous avez saisi dans un textbox

# Exercice 4 Composant

Ecrire un composant avec 2 bouton thumb up et thumb down et qui permet de compte le nombre de fois l’un des deux bouton a été cliqué

# Exercice 5 Composant Parent Enfant

Créer une vue qui permet d’afficher une liste

d’utilisateurs avec les valeurs suivantes: nom prenom

Un sous composant detail contact avec le numéro de téléphone l’email et une liste d’un sous composant les adresses.

# Exercice 6 Composant dynamique

Faire 2 bouton radio l’un avec le libellé voiture l’autre avec le libellé vélo

Creer 2 composants différents qui vont être afficher en fonction du bouton radio sélectionné.

Si on sélectionne voiture le composant affichera les questions suivantes dans un formulaire de saisie:

Type de véhicule

Manuel ou automatique

Nombre de chevaux

Consommation essence

Si c’est vélo même chose pour ces questions:

Type de velo

Taille du velo

Type de frein

Epaisseur des roues

On affichera les informations saisies à l'écran après que l’utilisateur est clique sur un bouton valider.

# Exercice 7 Transition

Créer un bouton qui change de valeur pour afficher ou masquer un paragraphe plus bas dans la page.

# Exercice 8 Transition

Créer une page qui permet d’afficher un modal d’information sur les informations d’ouverture et de fermeture d’une station de ski exemple:

Lundi-mercredi 8h30-16

Jeudi-samedi 10h-22h

Dimanche fermé

Exercice Interface Riche

## Exercice 1 - Props

Créer un composant qui prends une props valeur qui permet d’afficher un titre entre des balises h1 “Nouveau titre”

## Exercice 2 - Props

Créer un composant qui prends une props valeur qui permet d’afficher un titre entre des balises h1 ce titre sera saisi dans un input text par l’utilisateur

## Exercice 3 - Validator Props

Reprendre l’exercice précédent et ajouter un validateur de la propriété qui s’assure que le champs ne prends que des lettres. Vous pouvez utiliser l'instruction suivante:

/^[a-zA-Z]+$/.exec(valeur)

## Exercice 4 - slot

Faire un composant que vous remplirez avec 4 slot nommés de votre choix

## Exercice 5 - Watch

Faire 3 inputs un pour kilometre un pour mètre et un pour centimetre.

Lorsque l’on change la valeur de l’un des input on la conversion qui s’affiche dans les autres input grâce à l’attribut watch.

## Exercice 6 - Composant asynchrone

Faire un composant qui affiche le contenu d’une page d’accueil que vous chargerez de façon asynchrone au bout de 10 secondes.

On affichera le message “chargement en cours” en attendant son chargement.

Exercice Interface Riche 2

Pour cette pratique nous allons faire les différents codes dont nous aurons besoin dans un application VueJS.

Nous allons faire différentes pages indépendantes que nous allons réutiliser par la suite pour faire une application. Cela va nous permettre de revoir les différents concepts que nous avons vus jusqu'à maintenant.

Vous allez donc choisir un type d’entreprise que vous souhaitez représenter dans votre application. Vous pourrez lui donner un nom fictif et éventuellement un logo.

Notre application va se composer des pages suivantes:

Une page d’accueil

Qui contient une description de bienvenue par rapport à votre compagnie fictif et des points forts de celle-ci

D’une image

Une page a propos

Cette page sera faite sous la forme d’un modal qui s'affiche à termes sur la page en cours. Elle contiendra une description de la compagnie les informations de la compagnie taille domaine d'activité etc...

Une page de contact

Cette page contient les champs suivants:

Un champs email

Un champ sujet un dropdown de différents sujet déjà prédéfini que l’on aura

entre dans un attribut de type tableau dans notre composant et que l’on affichera grâce à une directive v-for.

Un champ text area. On s’assurera que chaque champ est rempli avant la soumission du formulaire et que le champ email respecte le pattern d’un email [a-z]@[a-z].[a-z]

On ajoutera également la possibilité dans l’application de créer un compte pour l’utilisateur.

Il faudra donc 3 autres pages

Une page de souscription qui contiendra les champs suivants.

Nom un champ input text

Prenom un champ input text

Adresse email un champ email

Confirmation de l’email un champ email

Mot de passe un champ mot de passe

Confirmation de mot de passe un champ mot de passe

Genre 3 radio bouton homme femme autre

Ville un drop down parmis des villes d’un tableau du composant et

Les champs nom prénom et adresse confirmation d’email et les deux champs de mot de passe seront obligatoires.

Une page de login

Une page d’accueil de l’utilisateur loger.

Avec un champ email et champs mot de passe

Encore une fois on vérifiera que les deux champs sont saisi avant d’accepter la validation

On fera une template avec des informations statiques d’un exemple des informations entrés lors de la souscription

Tout les pages citées seront des pages construites à partir d’un composant vue.js

Pour l’instant on inclura pas de routing (navigation) entre les pages on souhaites avoir le code de chacune de page de façon indépendante.

On pourra ajouter un framework comme bootstrap pour gérer l'esthétique de nos pages.