**VUE**

**Dure totale du module : 21H00**

Table des matières

[Objectifs pédagogiques : 3](#_Toc111661233)

[Notions / Concept de base de VUE 4](#_Toc111661234)

[2 Way DataBinding (EventBinding + DataBinding) directive v-model 4](#_Toc111661235)

[Exercice v-model 4](#_Toc111661236)

[Design Pattern MVC vs MVVM 7](#_Toc111661237)

# Objectifs pédagogiques :

Être capable de comprendre les enjeux des Frameworks

Être capable d'initialiser et paramétrer un projet grâce à un Frameworks front-end

Être capable de réaliser des interface utilisateur grâce à un Frameworks front-end

~~Être capable de mettre en place un système de Routing (url)~~

Être capable de gérer des requêtes HTTP (BDD)

Être capable de connecter une application à une base de données

# Notions / Concept de base de VUE

### 2 Way DataBinding (EventBinding + DataBinding) directive v-model

Précédemment nous avons abordé le fait de lier des attributs HTML et réagir à des évènements, dans certains cas d’utilisations impliquant des intput / textarea (souvent des formulaires), la directive   
v-model va permettre de lier directement (1 seule directive à utiliser) une variable (JS) (côté Model) avec le contenu (la value) d’un input / textarea (côté Vue).

### Exercice v-model

Mettre en place une application dans laquelle nous avons :   
°Un input qui utilise :

- v-on pour surveiller les touches du clavier et déclenche une fonction afficherUserName dans laquelle on capte l’évènement pour retrouver ce qui est tapé dans l’input (event.target.value) que l’on assigne à nameUser

- v-bind pour relier la value de cet input avec la variable nameUser

°Un affichage de la variable nameUser dans un paragraphe

°Un autre input sur lequel on utilise un v-model pour le lier à une variable userName

°Un affichage de la variable userName dans un paragraphe

[SOLUTION(GitHub)](https://github.com/jefff404/bookList-vue/tree/v1.8-exemple-v-model)

Fichier HTML :

<div id="monApp" class="container mt-4 mb-4 p-3">

<div class="mb-3">

<p>v-on + v-bind</p>

<input v-on:keyup="afficherNameUser" v-bind:value="nameUser" type="text" class="form-control" >

</div>

<div class="mb-3">

<p>{{nameUser}}</p>

</div>

<div class="mb-3">

<p>v-model</p>

<!-- modèle MVC vs MVVM -->

<input v-model="userName" type="text" class="form-control" >

</div>

<div class="mb-3">

<p>{{userName}}</p>

</div>

</div>

Avec des éléments HTML type inputs, v-model est plus direct.

Fichier JS :

const app = Vue.createApp({

data() {

return {

nameUser: '',userName: ''};

},

methods: {

afficherNameUser(event) {

this.nameUser = event.target.value;

}

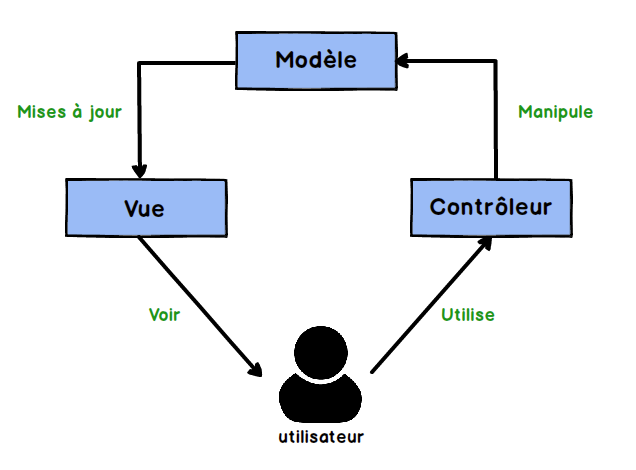
}

});

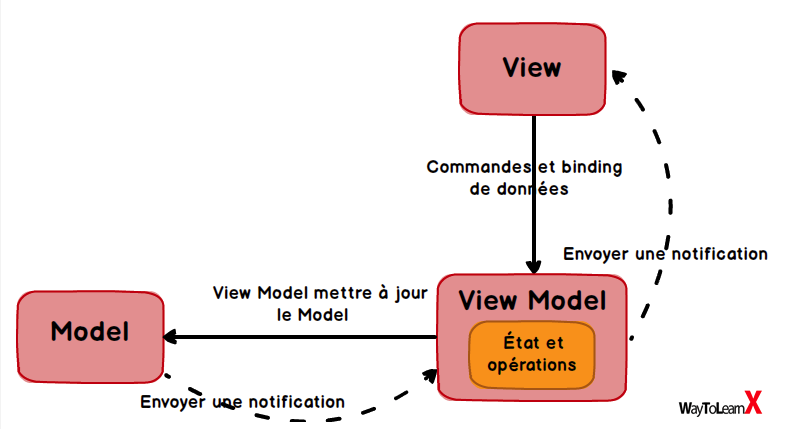
app.mount('#monApp');

### Design Pattern MVC vs MVVM

Cette directive v-model permet de mettre en évidence une approche différentes dans la construction d’application web, la plupart des applications web « classiques » utilisent un Design Pattern (architecture) MVC (Model, View, Controller) :



Avec v-model nous avons un Design Pattern MVVM :



<https://waytolearnx.com/2020/06/difference-entre-mvc-et-mvvm.html>