**VUE**

**Dure totale du module : 21H00**

Table des matières

[Objectifs pédagogiques : 3](#_Toc111663493)

[Notions / Concept de base de VUE 4](#_Toc111663494)

[Gestion des styles dynamiquement 4](#_Toc111663495)

[Style Dynamique (inline) 4](#_Toc111663496)

[Exercice : Style Dynamique (inline) 5](#_Toc111663497)

[Style Dynamique (class CSS) 8](#_Toc111663498)

[Exercice : Style Dynamique (class CSS) 8](#_Toc111663499)

# 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

## Gestion des styles dynamiquement

Une autre utilisation de la directive v-bind va permettre de gérer dynamiquement le style des éléments HTML. Techniquement tous les éléments HTML peuvent avoir un attribut style.

<p style="color:blue;">I am blue</p>

<p style="font-size:50px;">I am big</p>

### Style Dynamique (inline)

On peut utiliser v-bind pour lier l’attributs style d’un élément avec une variable ou une expression JS.

v-bind:style="{nomPropriétéCSS: duCodeJS}"

Par exemple : on peut définir 2 data côté JS

data() {

return {

activeColor: 'red',

fontSize: 30

}

}

Et les lier au style (plus précisément aux propriétés CSS color et fontSize).

<div :style="{ color: activeColor, fontSize: fontSize + 'px' }"></div>

### Exercice : Style Dynamique (inline)

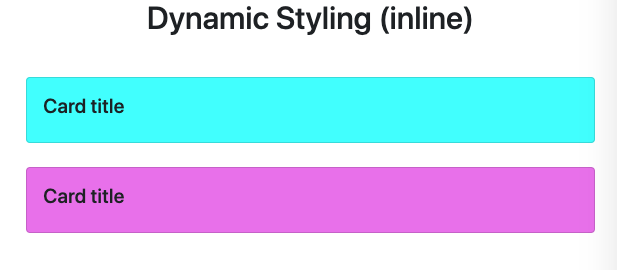
Dans une app Vue le click sur une div va appliquer un style :

Côté JS :

* On a 2 variables selectCard1 et selectCard2 par exemple, initialisées à false
* 1 fonction selectionCard qui prend en parmètre uneCard, dans laquelle on va faire :
  + SI uneCard égal 1 on passe selectCard1 à true
  + SI uneCard égal 2 on passe selectCard2 à true
  + !!! Bonus : faire en sorte : quand on reclick sur la div cela rechange la couleur

Côté HTML :

* Mettre en place 2 div (type card, box,etc..)
* Sur la 1ère div :
  + on réagit au click et déclenche la fonction selectionCard en passant 1 en paramètre
  + avec un v-bind de style, modifier le backgroundColor, dans une condition ternaire :
    - si selectCard1 est définit on retourne ‘red’ sinon ‘blue’
* Sur la 2ème div
  + on réagit au click et déclenche la fonction selectionCard en passant 2 en paramètre
  + avec un v-bind de style, modifier le backgroundColor, dans une condition ternaire :
    - si selectCard2 est définit on retourne ‘green’ sinon ‘violet’



[SOLUTION](https://github.com/jefff404/bookList-vue/tree/v1.12-dynamic-styling(inline))

Fichier HTML :

<div id="monApp">

<div class="row row-cols-1 row-cols-md-3 g-4 m-3">

<div class="col">

<div class="card" @click="selectionCard(1)"

:style="{backgroundColor: selectCard1 ? 'red':'aqua' }">

<div class="card-body">

<h5 class="card-title">Card title</h5>

</div>

</div>

</div>

<div class="col">

<div class="card" @click="selectionCard(2)"

:style="{backgroundColor: selectCard2 ? 'green':'violet' }">

<div class="card-body">

<h5 class="card-title">Card title</h5>

</div>

</div>

</div>

</div>

</div>

Fichier JS :

const app = Vue.createApp({

data() {

return { selectCard1:false,selectCard2:false };

},

methods: {

selectionCard(uneCard){

if(uneCard ===1){

this.selectCard1=true;

}

if(uneCard ===2){

this.selectCard2=true;

}

}

},});app.mount('#monApp');

D’autres cas d’utilisation de v-bind + style, on peut relier à l’attribut style d’un élément HTML, un objet JS contenant des propriétés CSS (c’est une utilisation très similaire à du CSS au final, à voir…)

Petit Rappel : En JS on manipule des propriété CSS avec la notation CamelCase tandis qu’en CSS normal on utilise la notation kebab-case (font-size en CSS vs fontSize en JS)

data() {

return {

styleObject: {

color: 'red',

fontSize: '13px'

}

}

}

<div :style="styleObject"></div>

### Style Dynamique (class CSS)

Si nous avons beaucoup de modification CSS à appliquer dynamiquement sur nos éléments, il sera plus pratique de passer par des classes que nous définissons au préalable en CSS qui seront activées ou non (true, false)

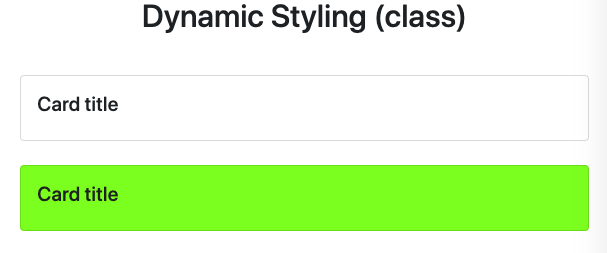
v-bind:class="{nomDeMaClasse: duCodeJS(true / false), nomDemaSecondeClasse :false}"

<div v-bind:class="{nomDeMaClasse: true, nomDeMaSecondeClasse :false}"></div>

### Exercice : Style Dynamique (class CSS)

Dans une app Vue, mettre en place 2 div (type card, box,etc..) :

* Quand on click sur la 1ère div cela applique une class CSS si une variable selectCard1 est définie
* Quand on click sur la 2ère div cela applique une autre class CSS si une variable selectCard2 est définie



[SOLUTION(GitHub)](https://github.com/jefff404/bookList-vue/tree/v1.13-dynamic-styling-class)

Fichier CSS ou balise <style> :

<style>

.class1{

background-color: mediumslateblue;

}

.class2{

background-color: chartreuse;

}

</style>

Fichier JS :

const app = Vue.createApp({

data() {

return {

selectCard1:false,

selectCard2:false

};

},

methods: {

selectionCard(uneCard){

if(uneCard ===1){

// this.selectCard1=true;

this.selectCard1=!this.selectCard1;

}

if(uneCard ===2){

// this.selectCard2=true;

this.selectCard2=!this.selectCard2;

}

}

},

});

app.mount('#monApp');

Fichier HTML :

<div id="monApp">

<div class="row row-cols-1 row-cols-md-3 g-4 m-3">

<div class="col">

<div class="card" @click="selectionCard(1)"

:class="{class1: selectCard1}">

<div class="card-body">

<h5 class="card-title">Card title</h5>

</div>

</div>

</div>

<div class="col">

<div class="card" @click="selectionCard(2)"

:class="{ 'class2': selectCard2}">

<div class="card-body">

<h5 class="card-title">Card title</h5>

</div>

</div>

</div>

</div>

</div>

Autre alternative utilisant un array, dans le cas où l’on veut v-bind l’attribut classe en combinant une classe statique (une classe que nous avons déjà défini en CSS) et la modification dynamique d’une propriété de style.

Exemple : ou une classe CSS « maClasse » serait déjà dans un fichier CSS.

<div :class="['maClasse',{backgroundColor:selectCard1}]" @click="selectionCard(1)"></div>