Aprendiendo VUE con las clases de bluueweb en youtube

# Clase 1: Introducción

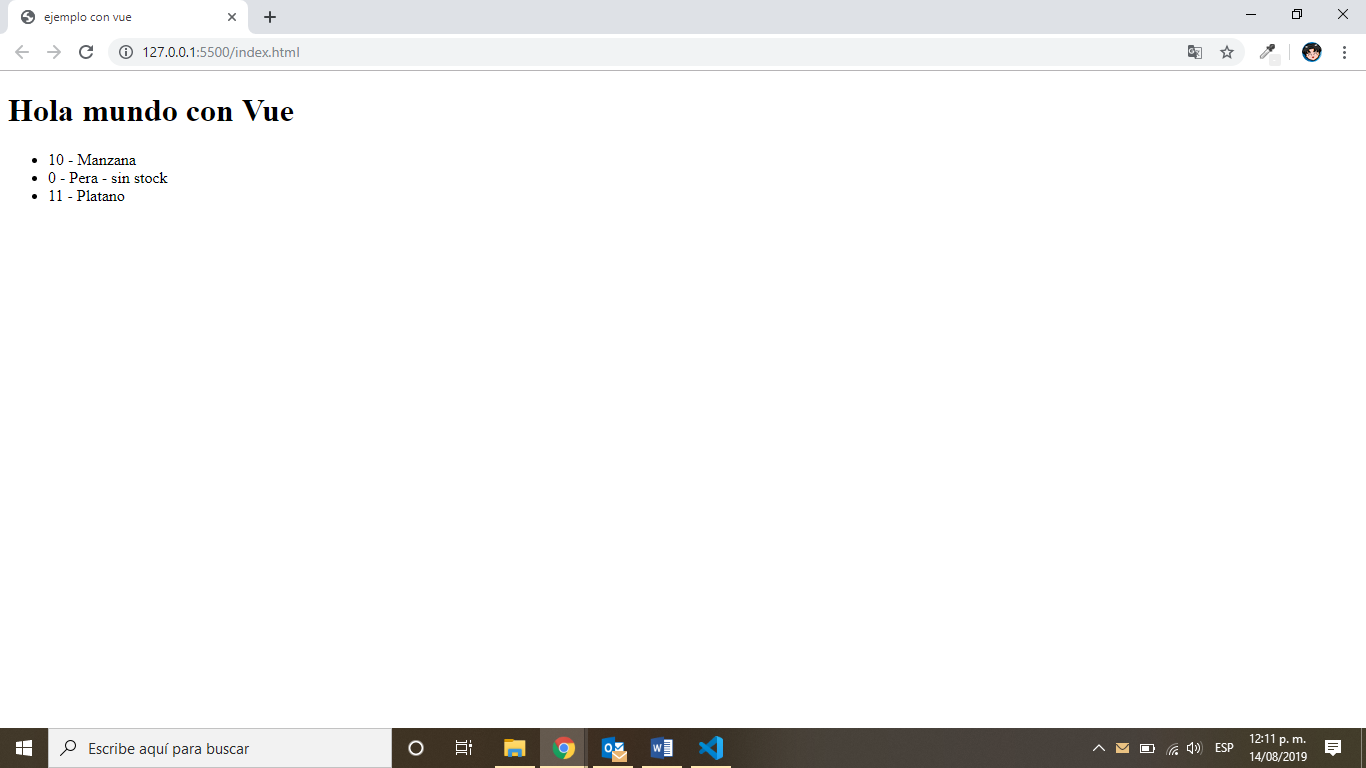
1. Para comenzar a usar vue copiamos el cdn para insértalo en nuestro html

<!-- development version, includes helpful console warnings -->

<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>

1. Tenemos que cerrar todo lo de vue en un container dentro del index.html
   1. <div id="app">
   2. <h1>{{titulo}}</h1>
   3. </div>
2. Para instanciar Vue dentro del archivo js/01.js
   1. const app = new Vue({
   2. el: '#app',
   3. data: {
   4. titulo: 'Hola mundo con Vue',
   5. frutas: [
   6. {nombre:'Manzana', cantidad:10},
   7. {nombre:'Pera', cantidad:0},
   8. {nombre:'Platano', cantidad:11}
   9. ]
   10. }
   11. })
3. El index.html quedaría de esta forma:
4. <!DOCTYPE html>
5. <html lang="en">
6. <head>
7. <meta charset="UTF-8">
8. <meta name="viewport" content="width=device-width, initial-scale=1.0">
9. <meta http-equiv="X-UA-Compatible" content="ie=edge">
10. <title>ejemplo con vue</title>
11. </head>
12. <body>
14. <div id="app">
15. <h1>{{titulo}}</h1>
16. <!-- COmo esun array debemos usar un siclo for y lo ponemos en una lista -->
17. <ul>
18. <li v-for="fruta of frutas">
19. {{fruta.cantidad}} - {{fruta.nombre}}
20. <span v-if="fruta.cantidad === 0"> - sin stock</span>
21. </li>
22. </ul>
23. </div>
24. <!-- development version, includes helpful console warnings -->
25. <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
26. <script src="js/01.js"></script>
27. </body>
28. </html>

Así quedaría la vista:



# Clase 2. Methods, v-model y evento click

1. Vamos ausar la etiqueta imput que es de los formularios
2. Para que el tecto sea detectdo por vue vamos a usar el v-model, con eso va ha existir un intercambio de datos desde nuestro index hacia nuestro javascript

<input type="text" v-model="nuevaFruta">

1. Ahora ben para agregarlo al array que tenemos en el index, recuerda que aparte del “el” y aparte del “data” tenemos Metódos. Dentro de esto vamos a crear un objeto y ponemos una función.

methods:{

agregarFruta () {

}

1. Vamos a probar todo esto haciendo un conole.log

console.log('diste click');

1. Ahora para conocer los eventos vamos a poner un botón en el index que diga agregar y ponemos el método de Click usando la @.

<button @click="agregarFruta">Agregar</button>

1. Ahora vamos a preparer todo para agregar la nueva fruta en nuestro array usando frutas.push(), literalmente lo que estamos haciendo es empujar la nueva fruta al array

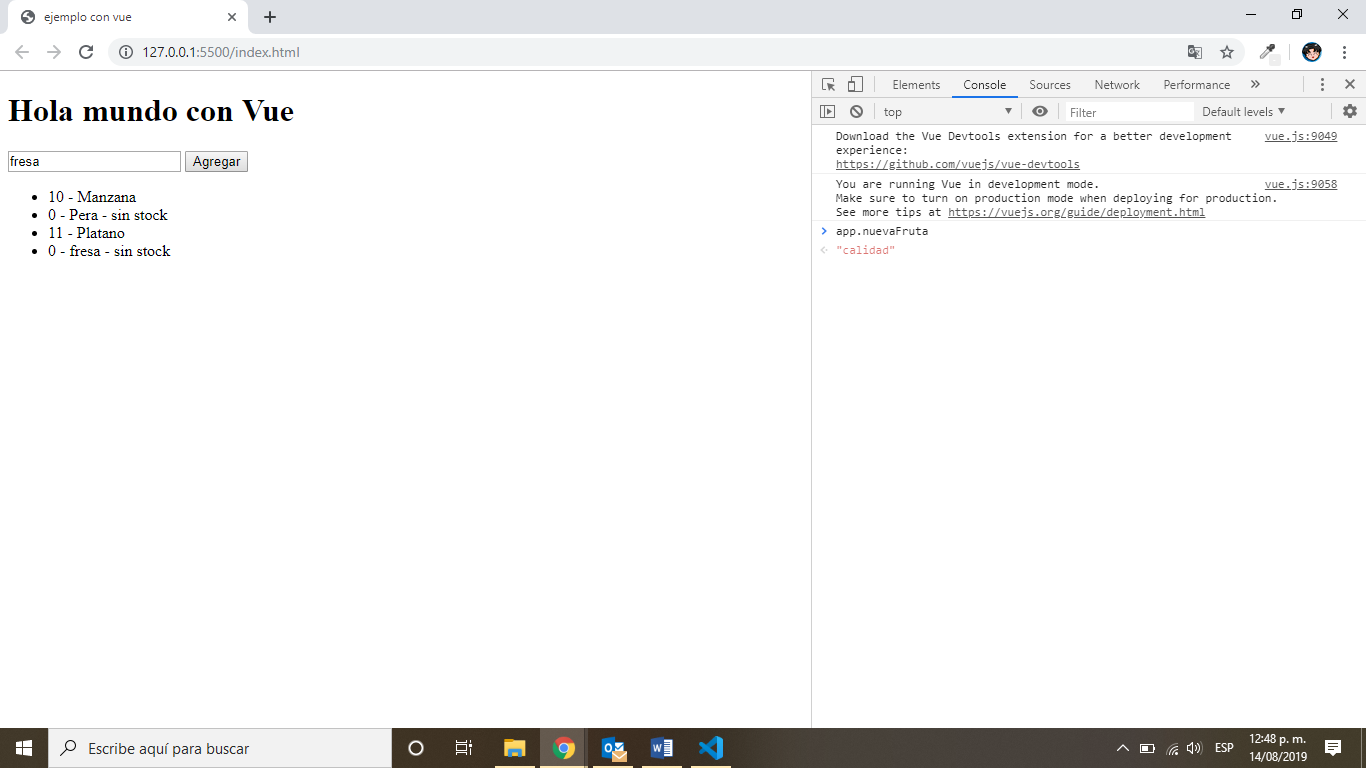
agregarFruta () {

this.frutas.push({

nombre: this.nuevaFruta, cantidad: 0

})

}



# Clase 3. Key-up, v-model y Computed

1. Para limpiar el contenido de nuestro imput vamos a sobre escribir la variable. Fuera de nuestro push() vamos a poner nuevaFruta = “”;

this.nuevaFruta = "";

1. Si queremos que al presionar el botón enter se ejecute la función agregarFruta podemos hacer lo siguiente dentro de nuestro input en el index.html

<input type="text" v-model="nuevaFruta" @keyup.enter="agregarFruta">

1. Para poder midificar el número de stock de cada fruto tenemos que transformar en imput el dato de fruta.cantidad

<input type="number" v-model="fruta.cantidad">

1. Si yo modifico la cantidad de la fruta que dice “sin stockautomaticamente desaparece esa etiqueta, sin embargo, si lo regreso a cero sigue sin aparecer, para arreglar esto. El problema recae es que el número que ponemos se refleja como string entonces para solucionar ese problema de decimios al v.model que va a ser de tipo numérico “v-model.number”

<input type="number" v-model.number="fruta.cantidad">

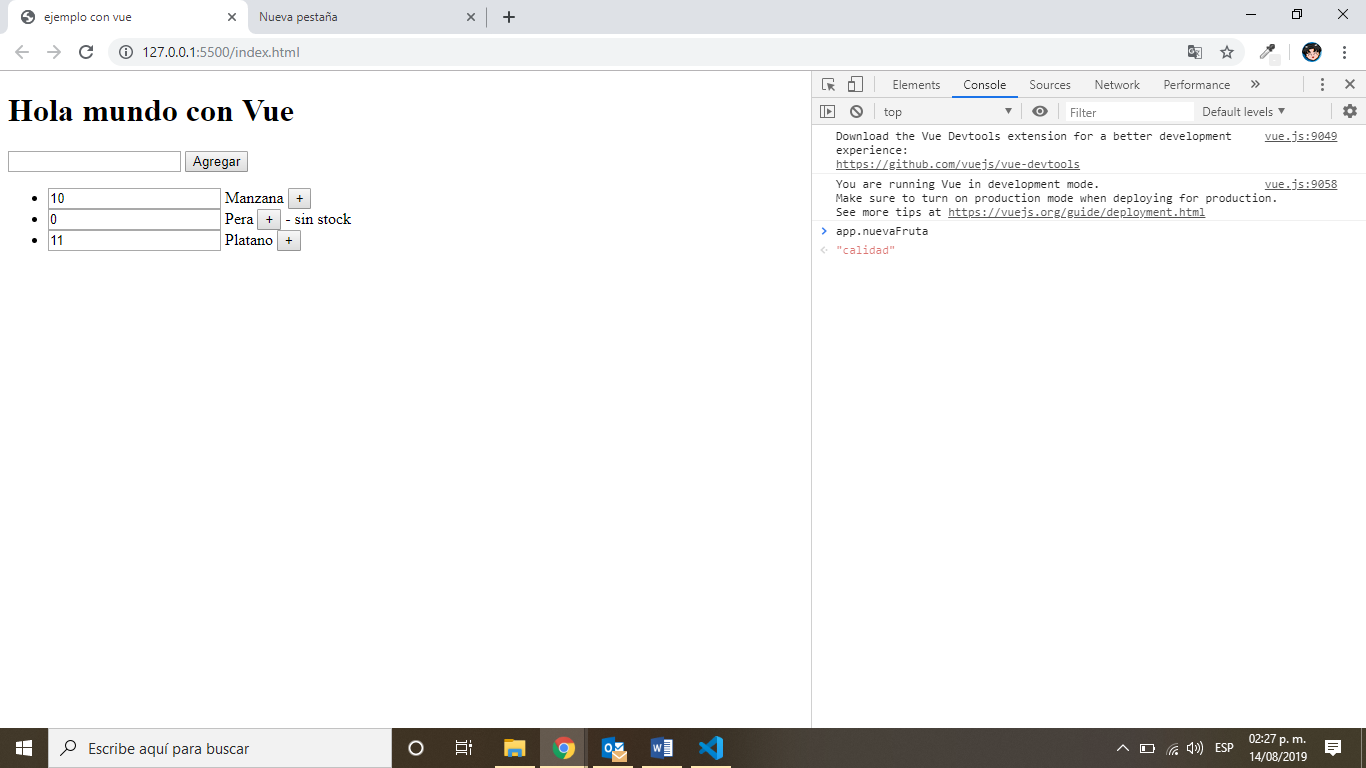
1. Ahora agregaremos un botón para poder sumar más frutas una por una de la siguiente manera

<input type="number" v-model.number="fruta.cantidad">

{{fruta.nombre}}

<button @click="fruta.cantidad++"></button>

1. Así quedaría la vista



1. Si ahora queremos sumar las cantidades, fuera del </ul> vamos aponer un h4

<h4>TOTAL: </h4>

1. Vamos a usar este nuevo elemento que se llama computed en nuestro archivo 01.js

computed: {

sumarFrutas () {

this.total = 0;

for (fruta of this.frutas){

this.total = this.total + fruta.cantidad;

}

return this.total;

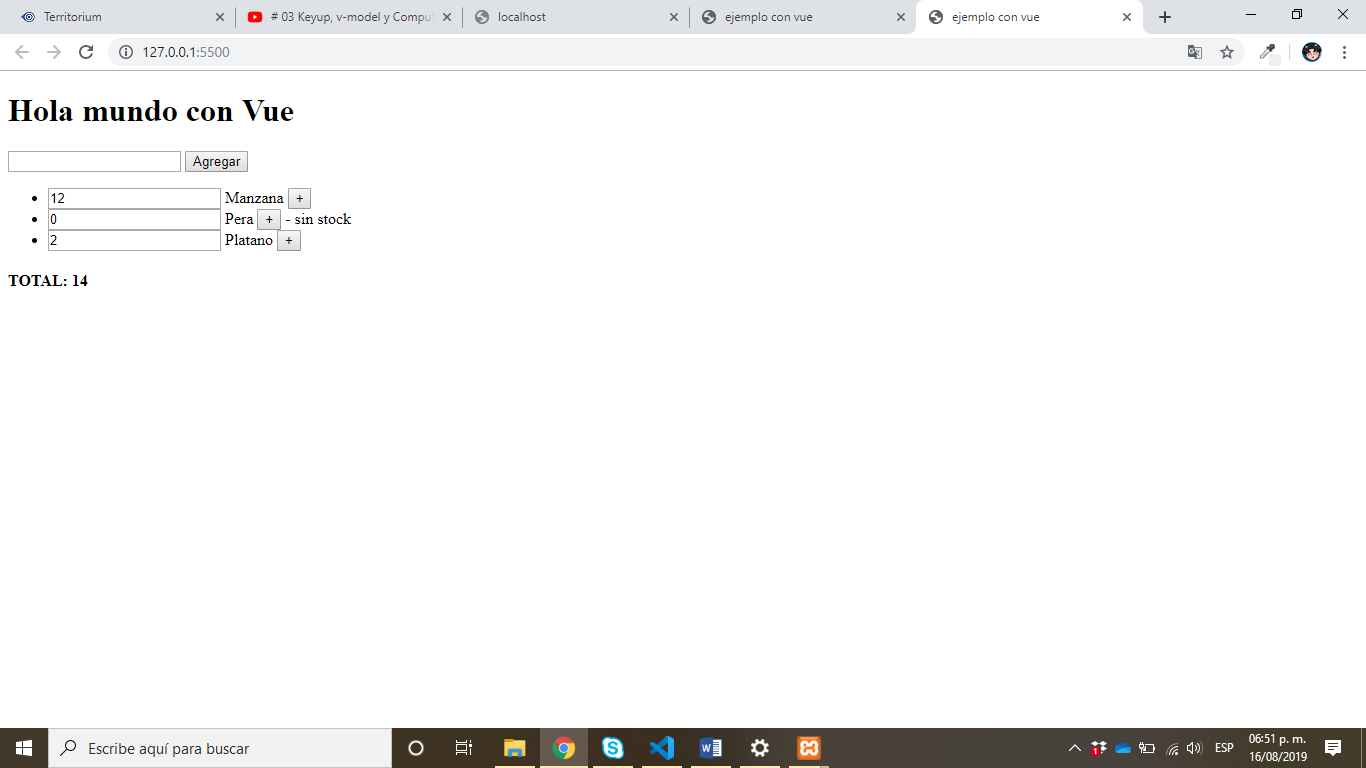
}

}

1. Cambiamos el index, solo la parte del total

<h4>TOTAL: {{sumarFrutas}} </h4>

1. Así quedaría la vista



# Clase 4. Clases (v-bind:class)

Copiamos los css de Bootstrap y vamos a trabar con vue

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

Es importante que recuerdes la clase app para poder seguir trabajando con vue, estaremos trabajando con el archivo 07.html

<div id="app" class="container mt-5">

<div class="bg-danger">

<p>Lorem ipsum dolor.</p>

</div>

</div>

Por el otro lado estaremos trabajando con javascript con su correspondiente 07.js

const app = new Vue({

el: '#app',

data: {

},

methods:{

}

})

Para poner lo mismo desde vue usamos :class=”’p-3’” para especificar las clases en vue hay que usar comillas simples dentro de las comillas dobles

class="bg-danger" :class="'p-3'"

Para poner más clases debemos hacer un array

<div class="bg-danger" :class="['p-3', 'text-white']">

Vamos a traer nuestro fondo desde vue en 07.js

const app = new Vue({

el: '#app',

data: {

fondo: 'bg-warning'

},

methods:{

}

})

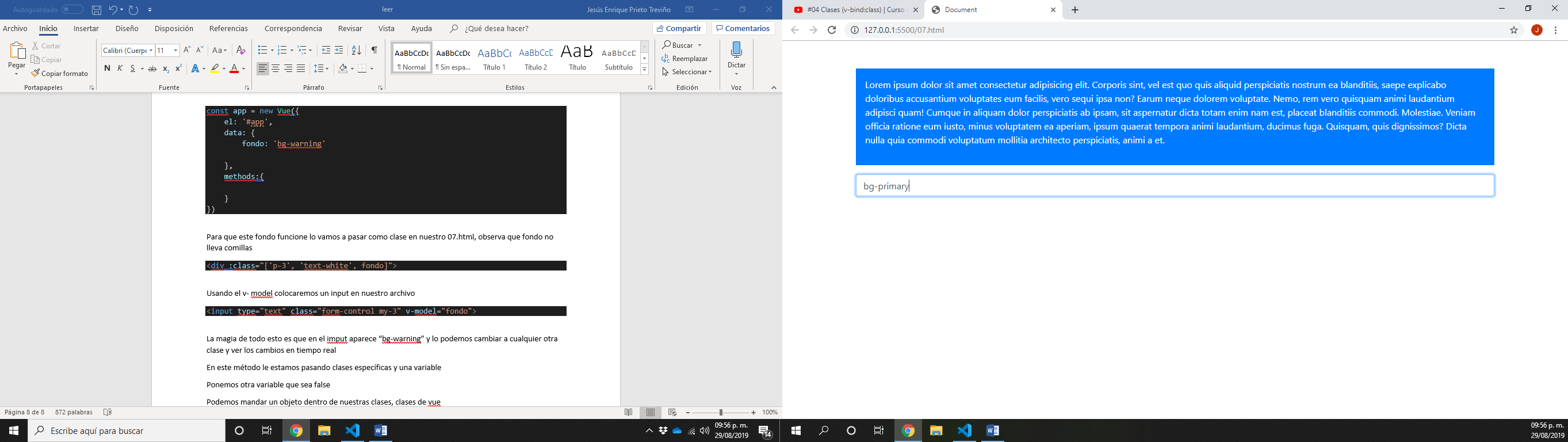
Para que este fondo funcione lo vamos a pasar como clase en nuestro 07.html, observa que fondo no lleva comillas

<div :class="['p-3', 'text-white', fondo]">

Usando el v- model colocaremos un input en nuestro archivo

<input type="text" class="form-control my-3" v-model="fondo">

En este método le estamos pasando clases específicas y una variable. La magia de todo esto es que en el imput aparece “bg-warning” y lo podemos cambiar a cualquier otra clase y ver los cambios en tiempo real



Ponemos otra variable que sea false

data: {

fondo: 'bg-warning',

color: false

},

Podemos mandar un objeto dentro de nuestras clases, clases de vue

<div class="text-white p-3" :class="{'bg-info' : color}">

Bg-info será colocoada si color es verdadero en el html

<div class="text-white p-3" :class="{'bg-info' : color, 'bg-success' : !color}">

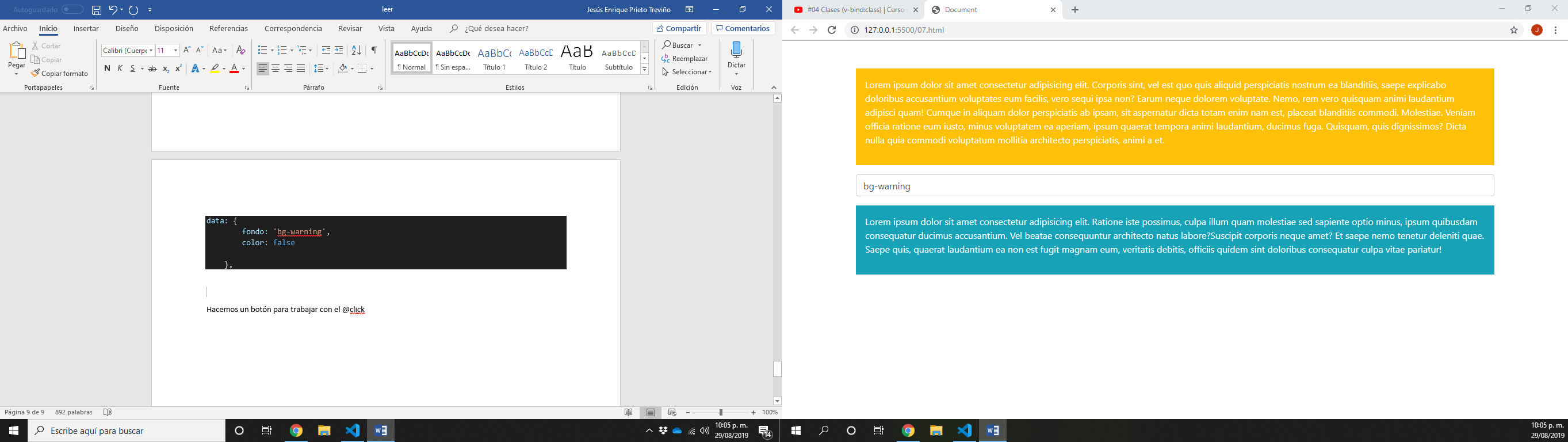
Este cambio lo hacemos en nuestro javascript

data: {

fondo: 'bg-warning',

color: false

},



Hacemos un botón para trabajar con el @click después del div para cambiar el color

<button class="btn btn-primary" @click="color = !color">cambiar</button>

El archivo final 07.html quedaría así:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

</head>

<body>

<div id="app" class="container mt-5">

<div :class="['p-3', 'text-white', fondo]">

<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Corporis sint, vel est quo quis aliquid perspiciatis nostrum ea blanditiis, saepe explicabo doloribus accusantium voluptates eum facilis, vero sequi ipsa non?

Earum neque dolorem voluptate. Nemo, rem vero quisquam animi laudantium adipisci quam! Cumque in aliquam dolor perspiciatis ab ipsam, sit aspernatur dicta totam enim nam est, placeat blanditiis commodi. Molestiae.

Veniam officia ratione eum iusto, minus voluptatem ea aperiam, ipsum quaerat tempora animi laudantium, ducimus fuga. Quisquam, quis dignissimos? Dicta nulla quia commodi voluptatum mollitia architecto perspiciatis, animi a et.</p>

</div>

<input type="text" class="form-control my-3" v-model="fondo">

<div class="text-white p-3" :class="{'bg-info' : color, 'bg-success' : !color}">

<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Ratione iste possimus, culpa illum quam molestiae sed sapiente optio minus, ipsum quibusdam consequatur ducimus accusantium. Vel beatae consequuntur architecto natus labore?Suscipit corporis neque amet? Et saepe nemo tenetur deleniti quae. Saepe quis, quaerat laudantium ea non est fugit magnam eum, veritatis debitis, officiis quidem sint doloribus consequatur culpa vitae pariatur!</p>

</div>

<button class="btn btn-primary" @click="color = !color">cambiar</button>

</div>

<!-- development version, includes helpful console warnings -->

<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>

<script src="js/07.js"></script>

</body>

</html>

Y el archivo final de 07.js quedaría así:

const app = new Vue({

el: '#app',

data: {

fondo: 'bg-warning',

color: false

},

methods:{

}

})

PRACTICA

# CRUD LocalStorage con Vue.js