VUE TRANSITIONS

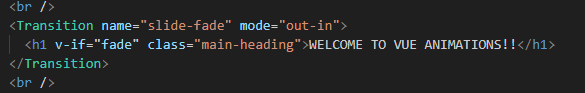
Las transiciones en Vue.js son una forma de agregar efectos de animación cuando se realizan cambios en el DOM, como la aparición, desaparición o cambio de elementos y componentes.

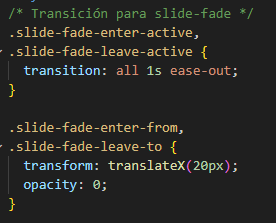
Vue.js proporciona un sistema de transiciones que se basa en la inclusión de clases CSS específicas en los elementos que están siendo animados. Este sistema se integra con propiedades de datos especiales y métodos del ciclo de vida de Vue para facilitar la creación de animaciones fluidas.

**Clases CSS:**

* + Vue.js añade clases CSS a los elementos que están siendo animados.
  + Las clases principales son **.v-enter**, **.v-enter-active**, **.v-exit**, y **.v-exit-active**.

**Ciclo de vida y Propiedades Especiales:**

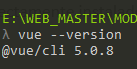
* + Para activar una transición, se pueden usar las propiedades especiales **v-enter**, **v-enter-active**, **v-exit**, y **v-exit-active** en los elementos Vue.
  + Estas propiedades se combinan con los métodos del ciclo de vida, como **beforeEnter**, **enter**, **afterEnter**, **beforeLeave**, **leave**, y **afterLeave**.
  + También se puede poner un nombre a una transición para atacarlo de manera especial, por ejemplo:

En esta transition se le añade el nombre slide-fade, lo que sustituye “v” en el css asociado al ciclo de vida del elemento h1. Por ejemplo: .v-enter pasa a ser .slide-fade-enter

PROYECTO EXPLICATIVO CON ROUTING Y ANIMACIONES EN VUE.js:

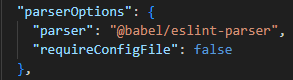
Requisitos y procedimiento para el proyecto base:

Asegurarse de tener instalado vue en el equipo, en caso contrario:

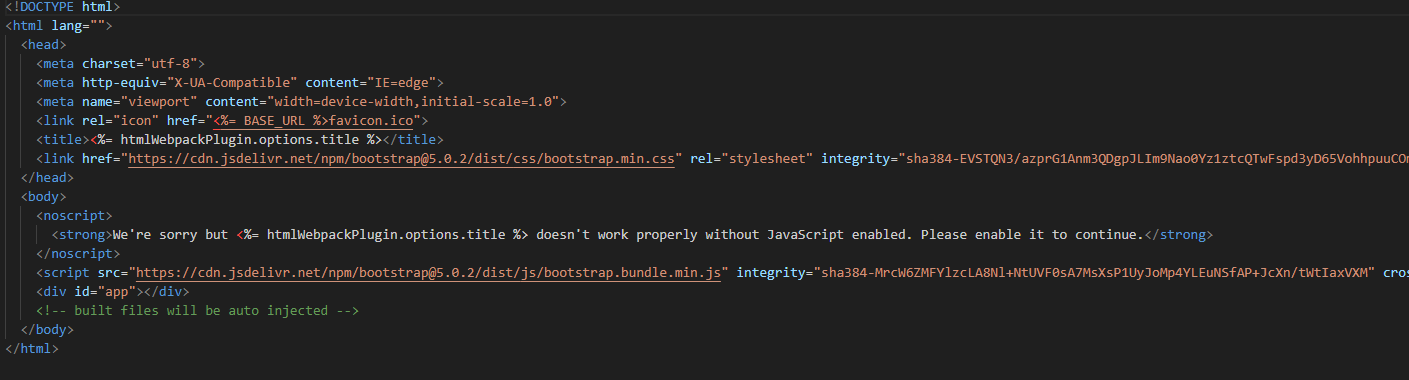
* **npm install -g @vue/cli 🡪** Instala vue en el equipo.
* **vue –version 🡪** Para comprobar si esta correctamente instalado.

En una carpeta o entorno habilitado, abrir un terminal de comandos e introducir los siguientes comandos:

* **vue create proyectotransitions 🡪** Para crear el proyecto de vue, este comando crea un repositorio de git automáticamente e instala las dependencias de npm que permiten emular el proyecto en local.
* **npm install vue-router@next –save 🡪** Instala y habilita el uso de routing en el proyecto.

En caso de experimentar errores en el editor por “BabelConfig”, introducir lo siguiente en el fichero package.json

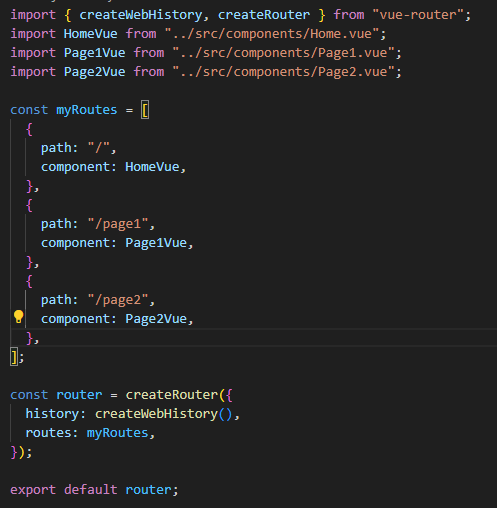
Inroducimos en el fichero: **public/index.html** las referencias a Bootstrap, librerías que uso para dar estilo fácilmente en este ejemplo:

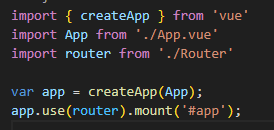


Para este ejemplo he creado los siguientes componentes de ejemplo, en la carpeta src/components:

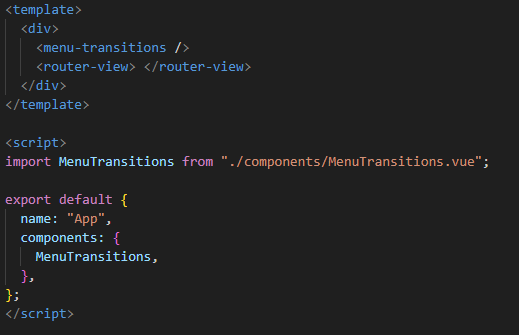
* Home.vue 🡪 Mi ejemplo hecho a mano sobre transiciones anidadas a partir de eventos
* MenuTransitions.vue 🡪 Navbar de vue que me permite la navegacion
* Page1 🡪 Ejemplo de usos
* Page2 🡪 Elemento para la navegación

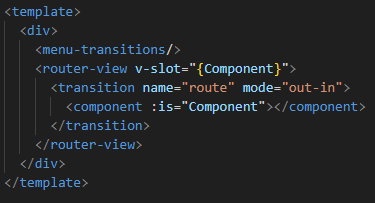
Lo primero que he hecho es añadir un router básico.

**Router.js:**

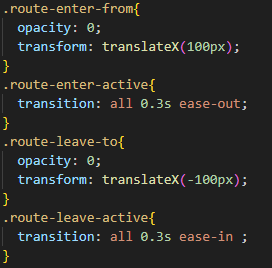
A partir de aquí hay que usar la variable router que estamos exportando en el momento de iniciar la App, para ello incluyo el siguiente código en src/main.js

Incluyendo el menú (que solo contiene un navbar de boostrap) y la vista de las rutas que acabamos de hacer, así quedaría App.vue:



Ahora bien, para habilitar una navegación más estética y dinámica, vamos a incluir una transición que se reproducirá en el momento en el que el componente que muestra el “router-view” cambie.

Le agregamos un nombre “route” y un modo: out-in, también podría ser in-out para que la animación se muestre al revés.

Ahora añadimos el css correspondiente al ciclo de vida de lo que en este ejemplo es “<component>”

Bien,