**EXAMEN VUE:**

1. **Introducción a Vue.js.**

Vue es un framework progresivo de JavaScript que se centra en la capa de visualización. Es ligero, rápido, y su curva de aprendizaje es menor en comparación con otros frameworks. Se utiliza para crear SPAs (Single Page Applications).

1. **Primera aplicación con Vue**

* **Modelo-Vista-Controlador**: Vue organiza la aplicación en un
* modelo (datos),
* vista (HTML) y
* controlador (instancia de Vue).
* Se recomienda usar Vue desde un CDN para iniciar, y su sintaxis básica incluye el uso de Vue.createApp() para inicializar aplicaciones.

1. **Clases dinámicas y propiedades computadas**

Explican cómo asignar clases dinámicas y utilizar propiedades computadas para lógica compleja. Estas propiedades se recomiendan por su eficiencia, ya que se almacenan en caché y se recalculan solo cuando es necesario.

1. **Herramientas para Vue**

Se destacan herramientas como Vue Devtools y extensiones para editores de código como Vetur y Vue Snippets.

1. **Directivas**

Directivas como v-bind, v-model, v-if, v-for y v-on extienden el comportamiento de elementos HTML. Se explican sus funcionalidades y cómo usarlas.

**Ampliación sobre las directivas de Vue**

**Directivas básicas**

1. **v-text**: Similar a la interpolación ({{ }}), inserta texto en un elemento. Si el dato cambia, se actualiza automáticamente.

html

<p v-text="mensaje"></p>

<!-- Equivalente a -->

<p>{{ mensaje }}</p>

1. **v-html**: Renderiza contenido HTML dentro de un elemento. A diferencia de v-text o interpolaciones, interpreta etiquetas HTML en lugar de escaparlas.

html

<div v-html="htmlString"></div>

1. **v-bind**: Vincula dinámicamente valores a atributos HTML. Se puede usar de forma abreviada con :.

html

<img :src="imagenUrl" alt="Descripción dinámica">

1. **v-model**: Permite el enlace bidireccional entre datos y elementos de formulario. Ideal para inputs, selects y checkboxes.

html

<input v-model="nombre">

1. **v-if, v-else-if, v-else**: Controlan la renderización condicional de elementos.

html

<p v-if="mostrar">Mostrando contenido</p>

<p v-else>No se muestra el contenido</p>

1. **v-show**: Similar a v-if, pero oculta elementos con display: none en lugar de eliminarlos del DOM.

html

Copiar código

<div v-show="mostrar">Siempre está en el DOM</div>

1. **v-for**: Renderiza elementos basados en arrays o rangos.

html

<ul>

<li v-for="(item, index) in lista" :key="index">{{ item }}</li>

</ul>

1. **v-once**: Renderiza un elemento solo una vez, ignorando actualizaciones posteriores de datos.

html

<h1 v-once>{{ titulo }}</h1>

1. **v-on**: Maneja eventos en elementos HTML. Puede abreviarse con @.

html

<button @click="saludar">Click aquí</button>

**Eventos comunes del ratón (v-on):**

1. **click**  
   Se dispara cuando el usuario hace clic en un elemento.

<button @click="accion">Click aquí</button>

1. **dblclick**  
   Se dispara al hacer doble clic en un elemento.

<button @dblclick="accionDoble">Doble click</button>

1. **mouseover**  
   Se dispara cuando el ratón pasa sobre un elemento.

<div @mouseover="destacar">Pasa el ratón por aquí</div>

1. **mouseout**  
   Se dispara cuando el ratón sale de un elemento.

<div @mouseout="quitarDestacado">Sal del área</div>

1. **mousemove**  
   Se dispara cuando el ratón se mueve dentro de un elemento.

<div @mousemove="moverMouse">Mueve el ratón aquí</div>

1. **contextmenu**  
   Se dispara al hacer clic derecho (menú contextual).

<div @contextmenu.prevent="mostrarMenu">Click derecho aquí</div>

**Eventos del teclado**

1. **keydown**  
   Se dispara cuando una tecla es presionada.

<input @keydown="teclaPresionada">

1. **keyup**  
   Se dispara cuando una tecla es soltada.

<input @keyup="teclaSoltada">

1. **keypress**  
   (Obsoleto en estándares modernos, pero aún usado en algunos contextos). Se dispara mientras una tecla está siendo presionada.

<input @keypress="teclaPulsada">

1. **Modificadores para teclas específicas**  
   Puedes usar nombres de teclas específicas con keydown o keyup:

<input @keydown.enter="enviar">

<input @keyup.esc="cancelar">

**Eventos de formulario**

1. **submit**  
   Se dispara al enviar un formulario.

<form @submit.prevent="enviarFormulario">

<button type="submit">Enviar</button>

</form>

1. **focus**  
   Se dispara cuando un elemento recibe foco.

<input @focus="focalizado">

1. **blur**  
   Se dispara cuando un elemento pierde foco.

<input @blur="perderFoco">

1. **change**  
   Se dispara cuando el valor de un elemento cambia.

<select @change="seleccionarOpcion">

<option value="1">Opción 1</option>

<option value="2">Opción 2</option>

</select>

1. **input**  
   Se dispara cada vez que cambia el valor de un campo de entrada.

<input @input="actualizarTexto">

**Eventos del sistema**

1. **load**  
   Se dispara cuando un recurso, como una imagen o una página, se carga completamente.

<img @load="imagenCargada" src="imagen.jpg">

1. **error**  
   Se dispara cuando hay un error al cargar un recurso.

<img @error="manejarError" src="imagen\_no\_existente.jpg">

1. **resize**  
   Se dispara cuando el tamaño de la ventana cambia.

<div @resize="ajustarVentana">Redimensiona la ventana</div>

1. **scroll**  
   Se dispara cuando el usuario desplaza la página o un contenedor.

<div @scroll="desplazado">Desplázame</div>

**Modificadores de eventos**

En muchos eventos, puedes usar **modificadores** para alterar el comportamiento predeterminado:

* **.prevent**: Evita la acción predeterminada (como recargar al enviar un formulario).

<a href="#" @click.prevent="noRecargar">Evitar recarga</a>

* **.stop**: Detiene la propagación del evento al elemento padre.

<button @click.stop="soloEsteElemento">Solo este botón</button>

* **.self**: Ejecuta el evento solo si ocurre en el elemento y no en sus hijos.

<div @click.self="soloDiv">Haz clic aquí</div>

* **.once**: Ejecuta el evento una sola vez.

<button @click.once="soloUnaVez">Solo una vez</button>