VUE ESSENTIALS
PÁGINA DE ATAJOS

FERNANDO HERRERA

EXPRESSIONS

```
<div id="app">
    <p>I have a {{ product }}</p>
    <p>{{ product + 's' }}</p>
    <p>{{ isWorking ? 'YES' : 'NO'}}</p>
    <p>{{ product.getSalePrice()}}</p>
</div>
```

DIRECTIVES

Elemento insertado/removido basado en veracidad

```
<p v-if="inStock">{{ product }}</p>
```

```
<p v-else-if="onSale">...</p>
<p v-else>...</p>
```

Alterna el display: ninguna propiedad CSS

```
<p v-show="showProductDetails">...</p>
```

Enlace de datos bidireccional

```
input v-model="firstName" >
```

`v-model.lazy="..."`	Sincroniza la entrada después del evento de cambio
`v-model.number="..."`	Siempre devuelve un número
`v-model.trim="..."`	Quita el espacio en blanco

LIST RENDERING

```
<li v-for="item in items"  :key="item.id">
  {{ item }}
</li>
```

key siempre se recomienda

Para acceder a la posición en la matriz

```
<li v-for="(item, index) in items">...
```

Para iterar a través de objetos

```
<li v-for="(value, key) in object">...
```

Usar v-for con un componente

```
<cart-product v-for="item in products"
              :product="item"
              :key="item.id">
```

Component Options:

Data	DOM	Lifecycle	Assets	Others
data	el	created	directives	inherit
props	template	mounted	elementDirectives	events
methods	replace	updated	filters	watch
computed		unmounted	components	mixins
		(Ver pagina siguiente)	transitions	name
			partials	

BINDING

```
<a v-bind:href="url">...</a>
```

shorthand →

```
<a :href="url">...</a>
```

Verdadero o falso va a agregar o remover el atributo:

```
<button :disabled="isButtonDisabled">...
```

Si isActive es veraz, la clase 'active' aparecerá:

```
<div :class="{ active: isActive }">...
```

Color de estilo establecido en el valor de activeColor

```
<div :style="{ color: activeColor }">
```

ACTIONS/EVENTS

Llama al método addToCart en el componente

```
<button v-on:click="addToCart">...
```

shorthand →

```
<button @click="addToCart">...
```

Se pueden pasar argumentos

```
<button @click="addToCart(product)">...
```

Para evitar el comportamiento predeterminado (por ejemplo, recargar la página)

```
<form @submit.prevent="addProduct">...
```

Solo dispara una vez

```
<img @mouseover.once="showImage">...
```

`.stop`	Detener toda la propagación de eventos
`.self`	Solo se activa si event.target es el elemento en sí

Ejemplo de entrada de teclado

```
<input @keyup.enter="submit">
```

Llame a onCopy cuando se presione control-c

```
<input @keyup.ctrl.c="onCopy">
```

Modificadores clave

`.tab`	`.space`	`.ctrl`	`.alt`
`.delete`	`.up`	`.left`	`.shift`
`.esc`	`.down`	`.right`	`.meta`

Modificadores de mouse

`.left`	`.right`	`.middle`

VUE ESSENTIALS
PÁGINA DE ATAJOS

FERNANDO HERRERA

ANATOMÍA DE COMPONENTES

```
Vue.component('my-component', {
 components: { Componentes que se pueden utilizar en la plantilla
  ProductComponent, ReviewComponent
},
props: { → Las propiedades que acepta el componente
  message: String,
  product: Object,
  email: {
    type: String,
    required: true,
    default: "none"
    validator: function (value) {

    } Debería devolver verdadero si el valor es válido
  }
},
data: function() { Debe ser una función
  return {
   firstName: 'Fernando',
   lastName: 'Herrera'
  }
},
computed: { Devolver valores en caché hasta que
  fullName: function () { las dependencias cambian
   return this.firstName + ' ' + this.lastName
  }
},
watch: {       Llamado cuando firstName cambia de valor
  firstName: function (value, oldValue) { ... }
},
 methods: { ... },
 template: '<span>{{ message }}</span>',
})          También puede utilizar comillas invertidas para varias líneas.
```

CUSTOM EVENTS

Use props(arriba) para pasar datos a componentes secundarios, eventos personalizados para pasar datos a elementos principales.

Establecer listener en el componente, dentro de su padre:

```
<button-counter v-on:incrementBy="incWithVal">
```

Dentro del componente principal:

```
methods: {
incWithVal: function (toAdd) { ... }
}
```

Plantilla interior de contador de botones:

```
this.$emit('incrementBy', 5)  Datos enviados al padre
```

Nombre del evento personalizado

LIFECYCLE HOOKS

beforeCreate	beforeUpdate	beforeUnmount
created	updated	unmounted
beforeMount	activated	errorCaptured
mounted	deactivated	renderTracked
		renderTriggered

USANDO UN SOLO SLOT

Template del componente:

```
<div>
  <h2>I'm a title</h2>
    <slot>
      Only displayed if no slot content
    </slot>
</div>
```

Uso de componente con datos para slot:

```
<my-component>
<p>This will go in the slot</p>
</my-component>
```

MULTIPLES SLOTS

Template del componente:

```
<div class="container">
   <header>
     <slot name="header"></slot>
   </header>
   <main>
     <slot>Default content</slot>
   </main>
   <footer>
     <slot name="footer"></slot>
   </footer>
</div>
```

Uso de componente con datos para slot:

```
<app-layout>
<h1 slot="header">Page title</h1>
<p>the main content.</p>
<p slot="footer">Contact info</p>
</app-layout>
```

LIBRERIAS QUE DEBERIAS CONOCER

Vue CLI
Interfaz de línea de comandos para un rápido desarrollo de Vue.

Vue Router
Navegación para una aplicación de una sola página.

Vue Devtools
Extensión del navegador para depurar aplicaciones Vue

Nuxt.js
Biblioteca para renderización del lado del servidor, división de código, recarga en caliente, generación estática y más

VUE 3 COMPOSITION API
PÁGINA DE ATAJOS

FERNANDO HERRERA

```
<template>
  <div>
    <p>Spaces Left: {{ spacesLeft }} out of {{ capacity }}</p>
    <h2>Attending</h2>
    <ul>
      <li v-for="(name, index) in attending" :key="index">
        {{ name }}
      </li>
    </ul>
    <button @click="increaseCapacity()">IncreaseCapacity</button>
  </div>
</template>
<script>
import { ref, computed } from "vue";
export default {
  setup() {
    const capacity = ref(4);
    const attending = ref(["Tim", "Bob", "Joe"]);
    const spacesLeft = computed(() => {
      return capacity.value - attending.value.length;
    });
    function increaseCapacity() {
      capacity.value++;
    }
    return { capacity, attending, spacesLeft, increaseCapacity };
  }
};
</script>
```

Utilice la API de composición cuando:

El componente es demasiado grande y debe estar organizado por lógica

Y / O

El código debe extraerse y reutilizarse a través de múltiples componentes, como una alternativa a los Mixins / Scoped Slots.

Y / O

El tipado de datos en TypeScript es importante.

Si usa Vue 2 con el complemento Composition API configurado:

```
import { ref, computed } from "@vue/composition-api";
```

Referencia reactiva
Envuelve primitivas en un objeto para rastrear cambios

Propiedad calculada

Acceda al valor de una referencia reactiva llamando `.value`

Métodos declarados como funciones

Le da a nuestra plantilla acceso a estos objetos y funciones.

TAMBIÉN SE PUEDE ESCRIBIR COMO:

```
import { reactive, computed, toRefs } from "vue";
export default {
  setup() {
    const event = reactive({
      capacity: 4,
      attending: ["Tim", "Bob", "Joe"],
      spacesLeft: computed(() => { return event.capacity -
event.attending.length; })
    });
    function increaseCapacity() {
      event.capacity++;
    }
    return { ...toRefs(event), increaseCapacity };
  }
};
```

Reactive toma un objeto y devuelve un objeto reactivo

Tenga en cuenta que no tenemos que usar `.value` ya que el objeto es reactivo

toRefs crea un objeto plano con referencias reactivas

VUE 3 COMPOSITION API
PÁGINA DE ATAJOS

FERNANDO HERRERA

ORGANIZAR POR FUNCIÓN

```
<template> ... </template>
<script>
export default {
  setup() {
    const productSearch = useSearch( )
    const resultSorting = useSorting({ })

return { productSearch, resultSorting }
   }
}
function useSearch(getResults) {

}
function useSorting({ input, options }) {

}
</script>
```

PARA EXTRAER CÓDIGO COMPARTIDO

```
<template> ... </template>
<script>
import useSearch from 'ause/search'
import useSorting from 'ause/sorting'
export default {
   setup() {
    const productSearch = useSearch( )
    const resultSorting = useSorting({ })

    return { product Search, resultSorting }
  }
 }
 </script>
```

use/search.js

```
export default function useSearch(getResults) {

}
```

use/sorting.js

```
export default function useSorting({ input, options }) {

}
```

El método setup ()

Llamado después del gancho beforeCreate y antes del hook creado.No tiene acceso a this.

Props

El primer argumento opcional de configuración

```
export default {
  props: {
    name: String
  },
  setup(props) {
    watch((). => {
       console.log( name is:  + props.name)
     })
   }
}
```

Los props son reactivos y se pueden ver

Context

El segundo argumento opcional de configuración

```
setup(props, context) {
  context.attrs;
  context.slots;
  context.emit;
}
```

expone propiedades previamente accedidas usando this

Life-cycle hooks

Declararlos dentro de la configuración

```
setup() {
  onMounted(() => { ... });
  onUpdated(() => { ... });
  onUnmounted(() => { ... });
}
```

En lugar de usar BeforeCreate o enlaces creados, simplemente escriba código o llame a funciones

Fuentes

devhints.io/vue, Vue CheatSheet
vuemastery.com/pdf/Vue-3-Cheat-Sheet.pdf, Vue Composition Api Cheat Sheet
http://vuejs.org/api/options