¿Qué conocimientos tenemos?

Primeros Pasos

Vue 3 CheatSheet

SINTAXIS DE PLANTILLA

Opciones de interpolación de texto

```
<span> {{ msg }} </span>
<span v-text="msg"></span>
```

Puedes usar expresiones JS

```
<span> {{ mg.reverse() }} </span>
```

DIRECTIVAS

- **v-if**, Pone el en DOM si es verdadero
- **v-else-if**, Como un condicional habitual
- **v-else**, Como un condicional habitual
- **v-show**, Conmuta el valor CSS de visualización
- **v-text**, Establece el texto interno
- **v-html**, Establece el HTML interno
- **v-for** Bucle a través de una matriz / obj
- **v-on o @**, Escucha eventos DOM
- **v-bind o :**, atributo de actualizaciones reactivas
- **v-model**, Enlace de datos bidireccional
- **v-once**, Establece val una vez; Nunca actualice

RENDERIZANDO CONDICIONAL

Agregar / quitar elemento de DOM

```
<div v-if="date == today"> ... </div>
<div v-else-if="! done"> ... </div>
<div v-else> ... </div>
```

Alterna la visualización de CSS en el lugar de editar DOM

```
<div v-show="date == today">...</div>
```

RENDERIZADO DE LISTA

Iteración básica sobre la lista

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

Iteración e índice de seguimiento

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

Iteración sobre los valores del objeto

```
<li v-for="(value, key) in object">
  {{ key }} : {{ value }}
</li>
```

ENLACE DE DIRECTIVAS

Enlace simple

```
<div v-bind:id = "objectI D"> ... </div>
<!-- SHORTHAND -->
<div :id = "objectID"> ... </div>
```

Enlace bidireccional

```
<input v-model="email" />
```

Usar objetos para enlanzar a clases / estilos

```
<input :class = "{error: hasError}" />
<input :style = "{margin: space + "px"}" />
```

Modificadores de entrada

```
.lazy en evento de cambio
.trim elimina los espacios en blanco adicionales
```

MANEJO DE EVENTOS

Captura un evento y llama a un método

```
<div v-on:click="count">Increase</div>
<!-- SHORTHAND -->
<div @click="count">Increase</div>
```

COMUNICACIÓN ENTRE COMPONENTES PADRE E HIJO

Utilice v-bind para pasar datos de padres a hijos y emita un evento personalizado para devolver los datos.

En el padre, vincularlistener datos y configurar el listener que actualizará

```
<my-component :msg="s" @update="s=$event"/>
```

En niño, enviar de vuelta usando emit (evento, datos)

```
  this.$emit("update", "hello world")
```

Componentes

¿Qué es un Single File Component?

Vue Single File Components (también conocido como archivos * .vue, abreviado como SFC) es un formato de archivo especial que nos permite encapsular la plantilla, la lógica y el estilo de un componente Vue en un solo archivo.

Ejemplo de un SFC

```
<script>
export default {
  data() {
    return {
      greeting: "Hello World!"
    }
  }
}
</script>

<template>
  <p class="greeting">{{ greeting }}</p>
</template>

<style>
.greeting {
  color: red;
  font-weight: bold;
}
</style>
```

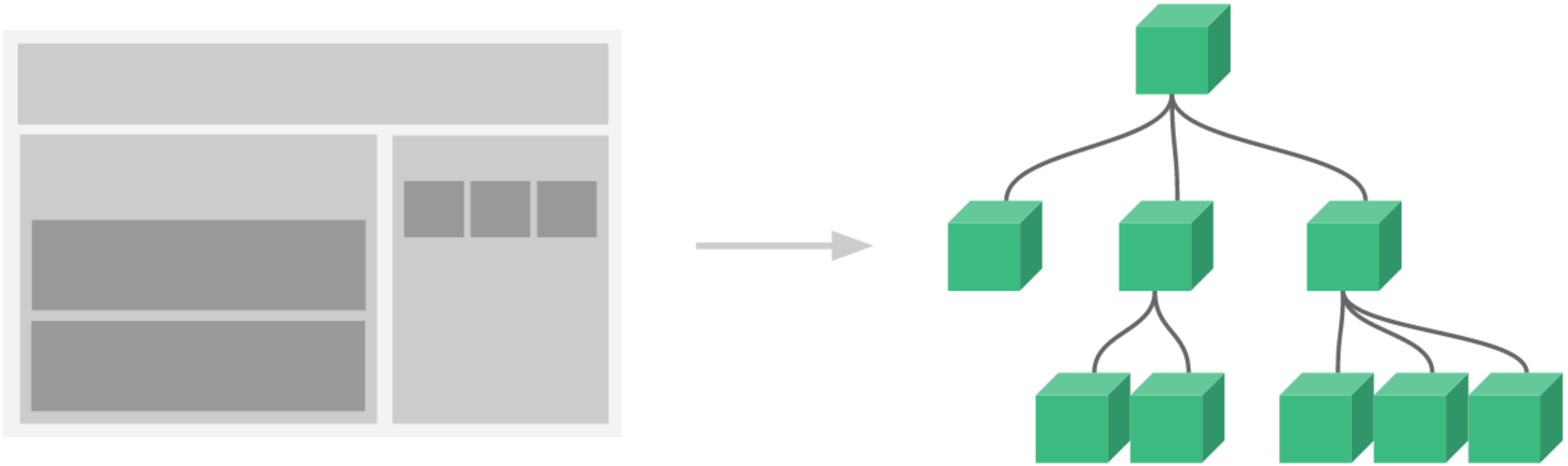
Ejemplo de un SFC

Como podemos ver, Vue SFC es una extensión natural del clásico trío de HTML, CSS y JavaScript. Cada archivo * .vue consta de tres tipos de bloques de lenguaje de nivel superior: `<template>` ,`<script>`y `<style>` :

- La sección `<script>` es un módulo estándar de JavaScript. Debería exportar una definición de componente de Vue como su exportación predeterminada.
- La sección `<template>` define la plantilla del componente.
- La sección `<style>` define CSS asociado con el componente.

Organización de componentes

Es común que una aplicación se organice en un árbol de componentes anidados:

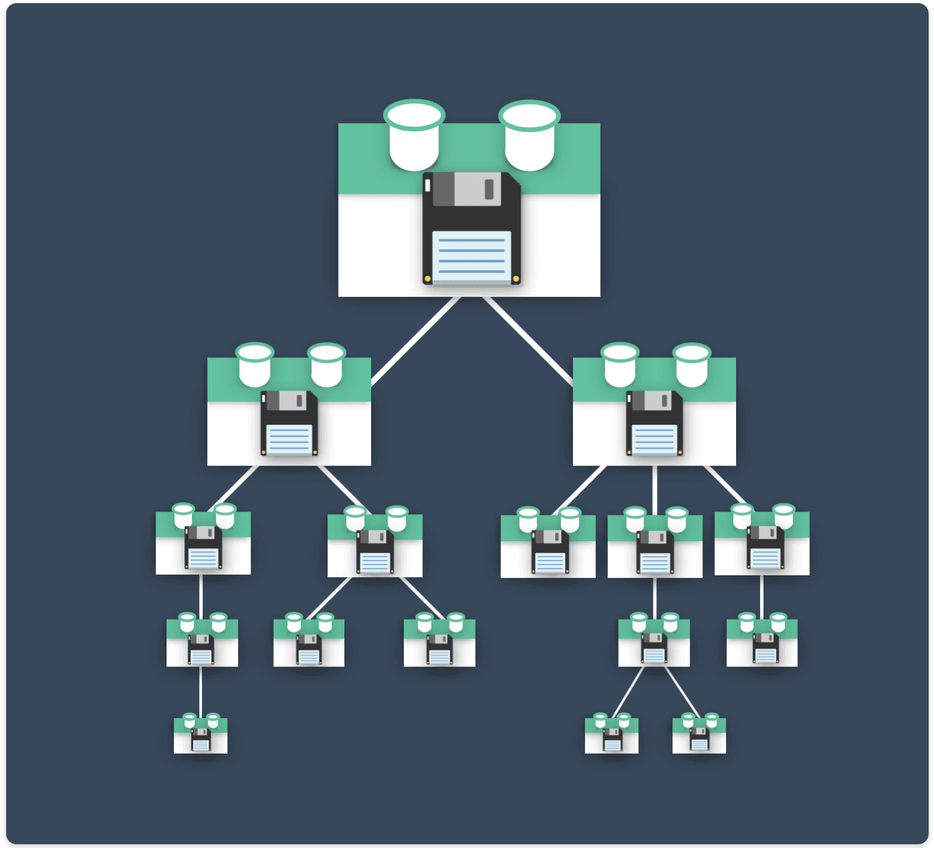


Por ejemplo, puede tener componentes para un encabezado, barra lateral y área de contenido, cada uno de los cuales normalmente contiene otros componentes para enlaces de navegación, publicaciones de blog, etc.

Gestión del estado entre componentes

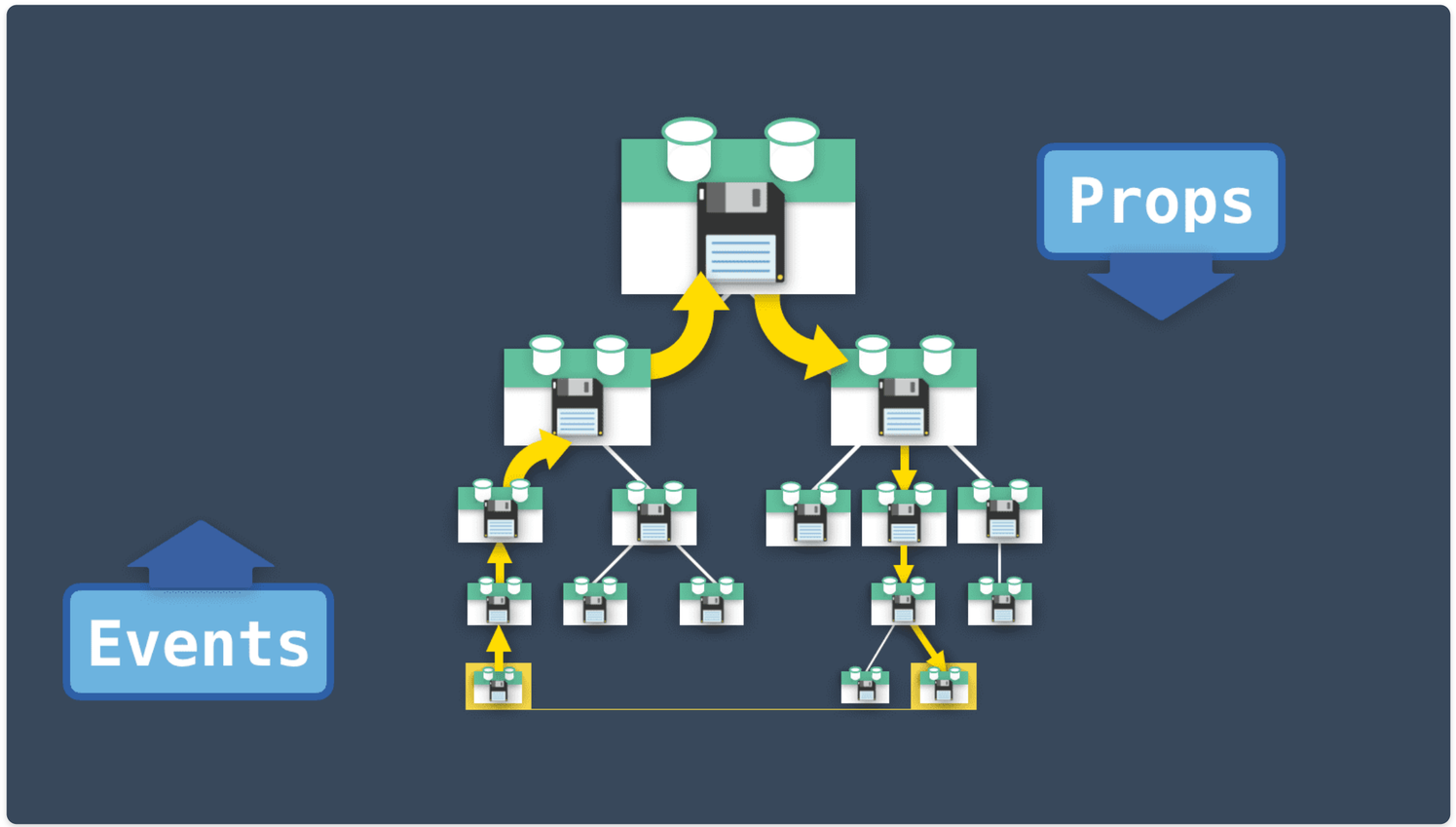
Gestión del estado

- Cuando hablamos de estado, nos referimos a los datos de los que dependen y procesan sus componentes. Cosas como publicaciones de blog, elementos de tareas pendientes, etc.
- Administrar el estado en una aplicación llena de componentes puede resultar difícil.



Gestión del estado

Si un componente cambia de estado y un pariente lejano también está usando ese mismo estado, necesitamos comunicar ese cambio. Existe la forma predeterminada de comunicar eventos y transmitir accesorios para compartir datos, pero eso puede volverse demasiado complicado.



¿Qué necesitamos?

- Necesitamos un lugar donde consolidar todo nuestros estados en un solo lugar.
- Una ubicación que contiene el estado actual de toda nuestra aplicación.
- Una fuente 100% confiable que pueda ser depurable.

¿Tenemos una solución? … si, Flux

¿Qué es flux?

Flux es un patrón para administrar el flujo de datos en su aplicación. El concepto más importante es que los datos fluyen en una dirección. A medida que avanzamos en esta guía, hablaremos sobre las diferentes partes de una aplicación Flux y mostraremos cómo forman ciclos unidireccionales por los que pueden fluir los datos.

Elementos de Flux:

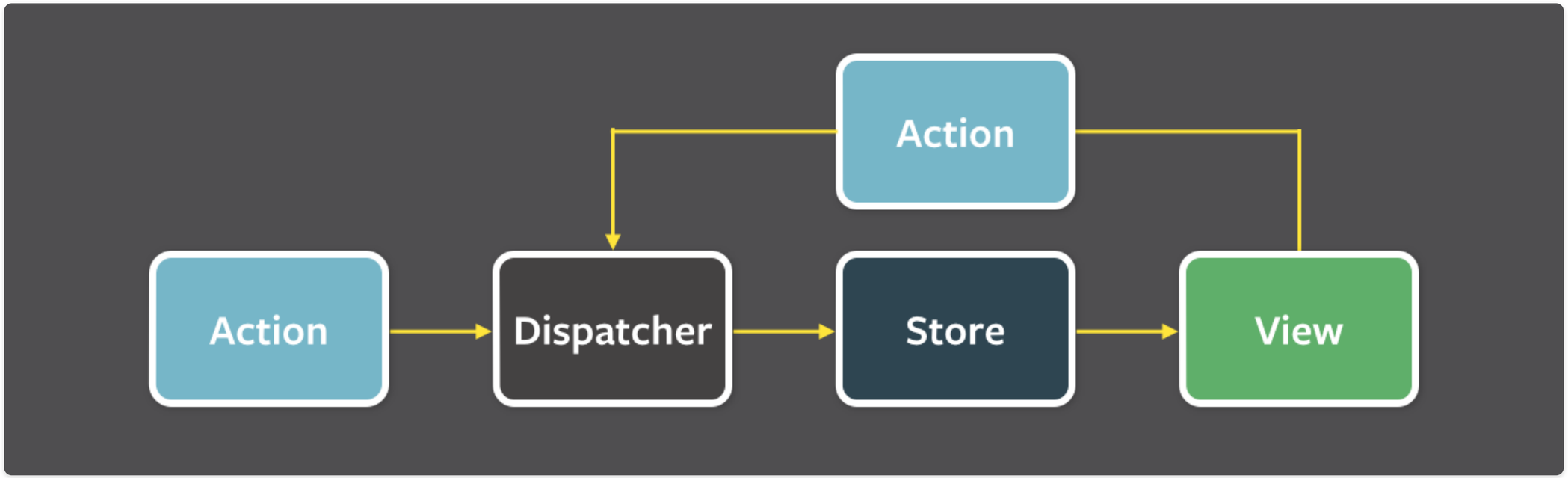
- Dispatcher
- Store
- Action
- View

Flux

¿Cúal el flujo de flux?

Podemos dividir las partes de Flux anteriores en un diagrama que describa cómo fluyen los datos a través del sistema.

- Las vistas envían Actions al Dispatcher.
- Dispatcher envía Actions a cada Store.
- Store envía datos a las vistas, (Desde otro punto de vista: View obtienen datos de Store).



¡Gracias por escuchar!
¿Preguntas?