

# Curso Vue.js + Ionic

## Sesión 1

Juan Searle. 13 junio 2022

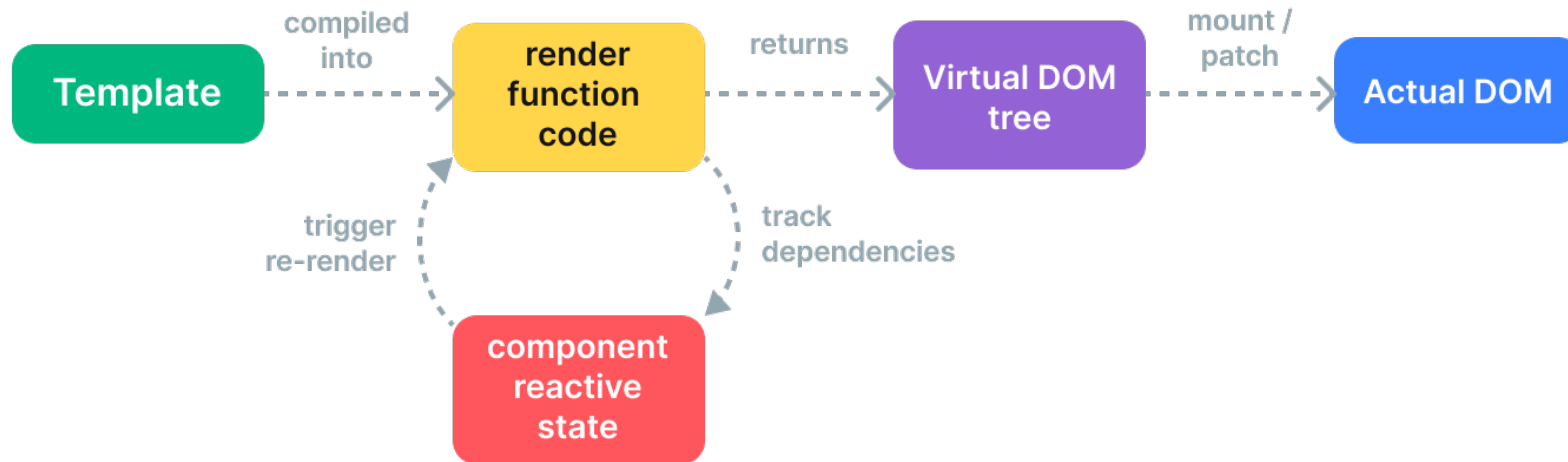
# Vue.js

1. Introducción a Vue.js
2. Interacción con el DOM
3. Componentes
4. Comunicación entre componentes
5. Uso avanzado de componentes
6. Directivas personalizadas



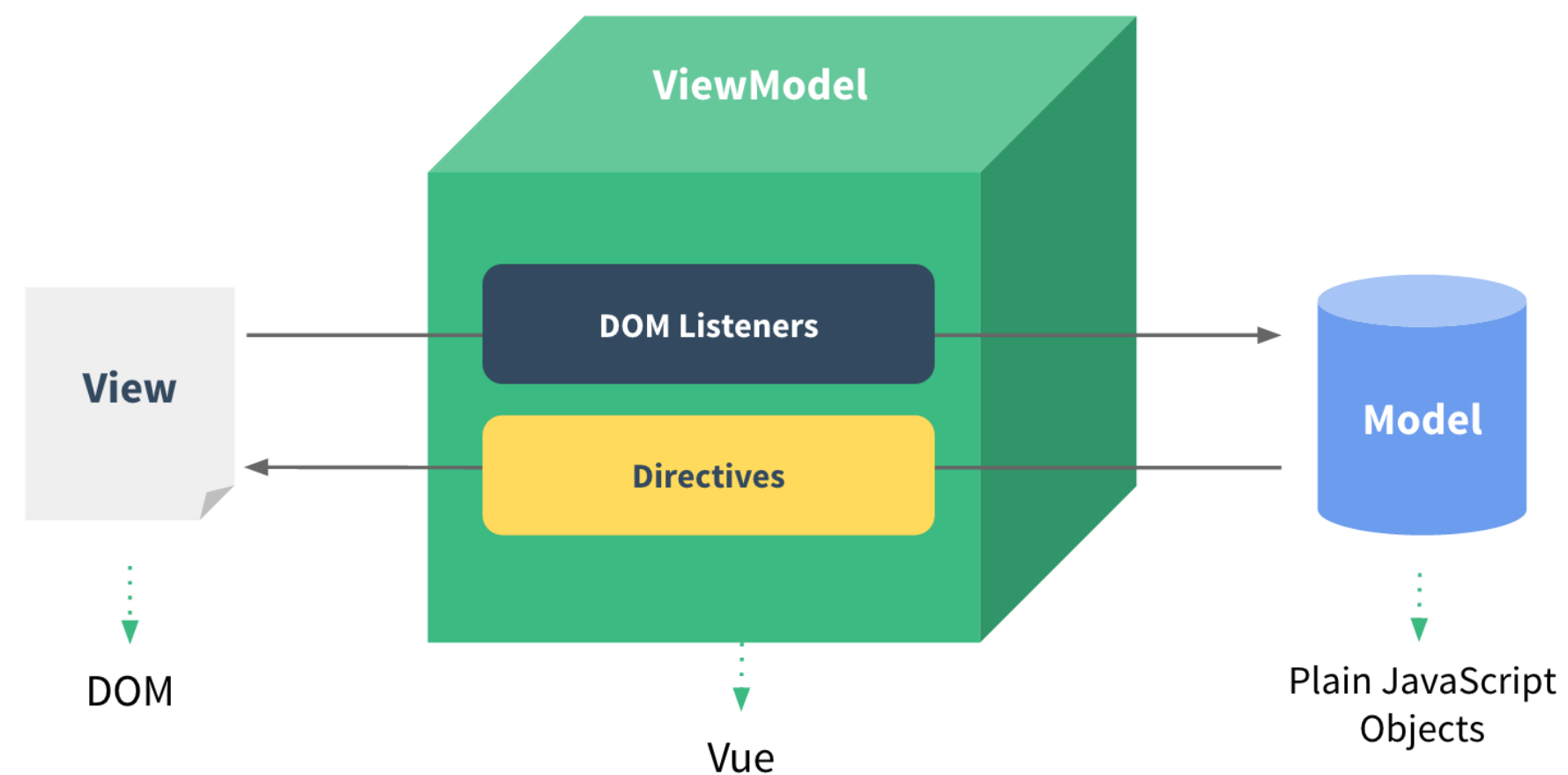
# 1. Introducción a Vue.js

Nociones elementales



# 1. Introducción a Vue.js

## Estructura básica



```
app.vue

<template>
  
  <MainComponent />
</template>

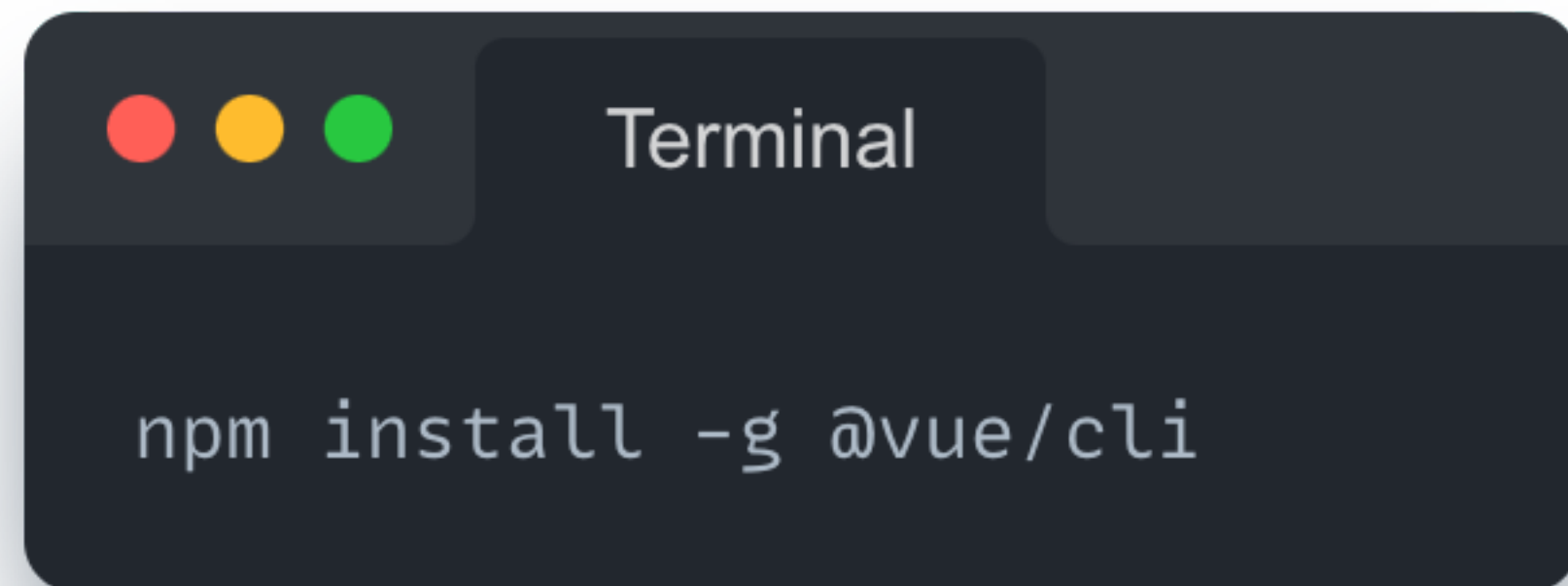
<script lang="ts">
  import { defineComponent } from 'vue';
  import MainComponent from '../components/MainComponent.vue';

  export default defineComponent({
    name: 'App',
    components: {
      MainComponent
    }
  });
</script>

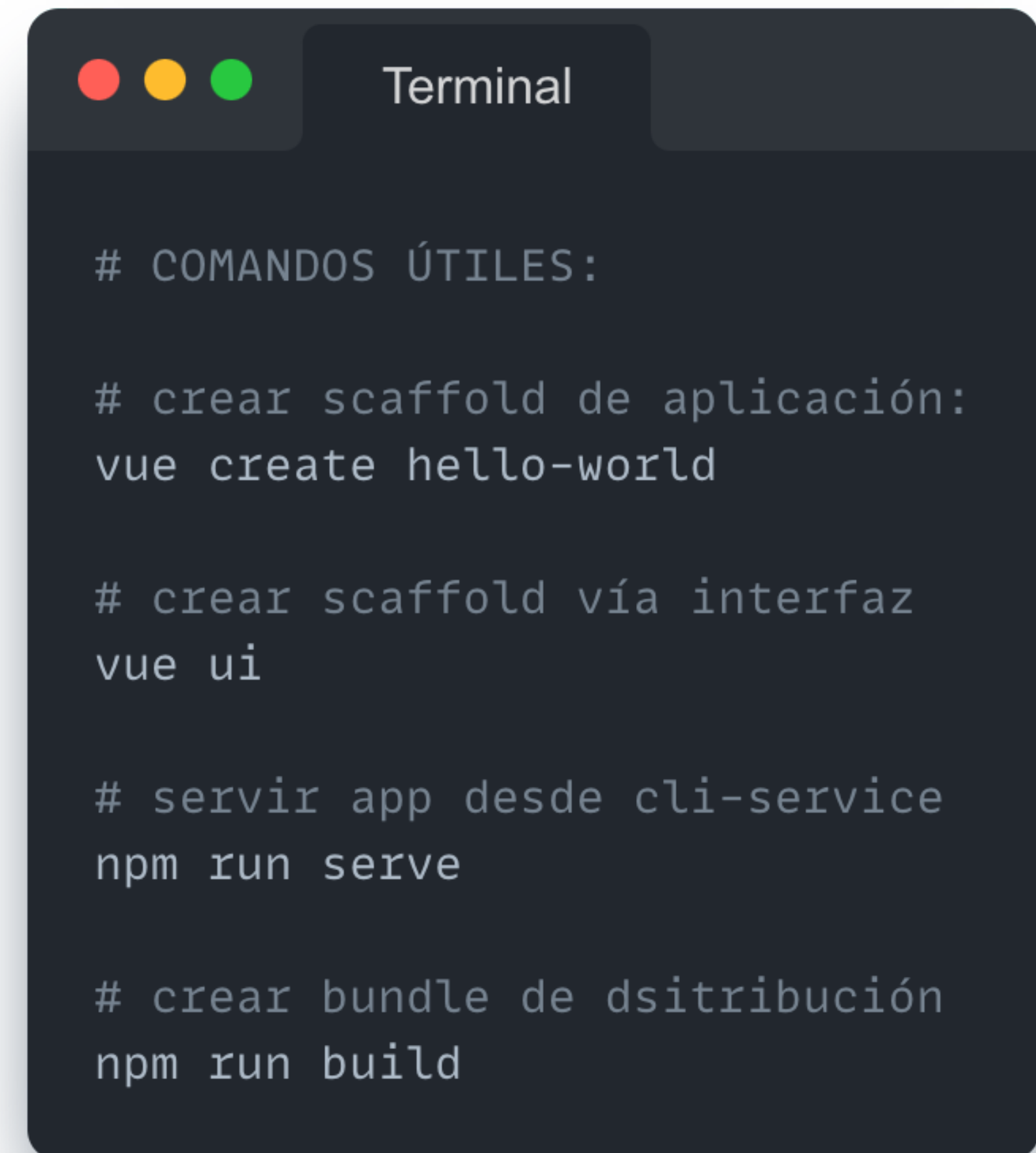
<style>
  #app {
    font-family: Avenir, Helvetica, Arial, sans-serif;
  }
</style>
```

# 1. Introducción a Vue.js

Vue CLI:



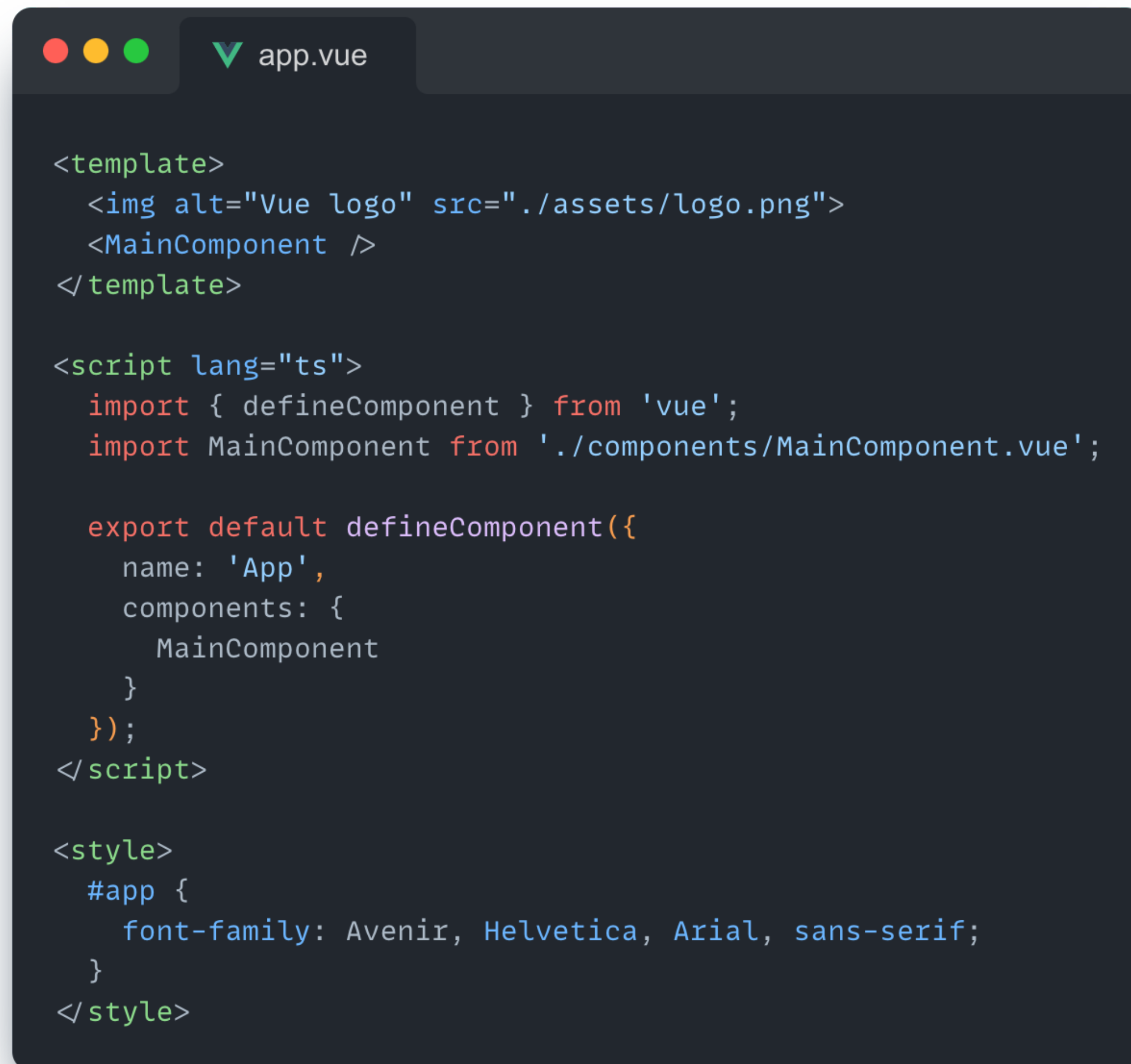
```
Terminal  
npm install -g @vue/cli
```



```
Terminal  
  
# COMANDOS ÚTILES:  
  
# crear scaffold de aplicación:  
vue create hello-world  
  
# crear scaffold vía interfaz  
vue ui  
  
# servir app desde cli-service  
npm run serve  
  
# crear bundle de distribución  
npm run build
```

# 2. Interacción con el DOM

La plantilla de Vue

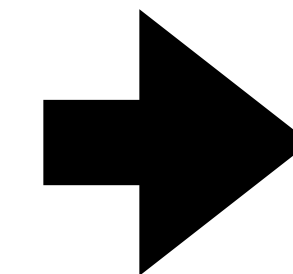


```
<template>
  
  <MainComponent />
</template>

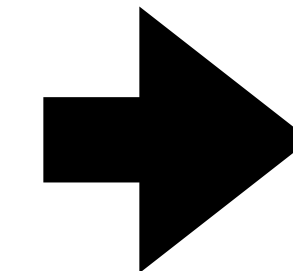
<script lang="ts">
  import { defineComponent } from 'vue';
  import MainComponent from '../components/MainComponent.vue';

  export default defineComponent({
    name: 'App',
    components: {
      MainComponent
    }
  });
</script>

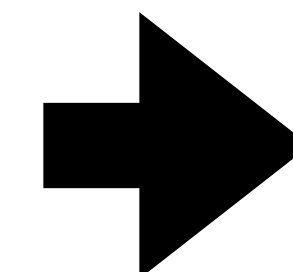
<style>
  #app {
    font-family: Avenir, Helvetica, Arial, sans-serif;
  }
</style>
```



Plantilla HTML



Script Typescript



Estilos CSS

# 2. Interacción con el DOM

## Binding

Normal

Abreviado

```
Mycomponent.vue

<template>
  <h2>Este es mi componente</h2>
  <p>{{number}}</p>
  <p v-bind:title="text"></p>
  <button v-on:click="buttonClicked">Haz click</button>
</template>

<script lang="ts">
import { defineComponent } from 'vue'

export default defineComponent({
  name: 'MyComponent',
  data(){
    return{
      number:0,
      text: 'Hola'
    }
  },
  methods:{
    buttonClicked(){
      this.number++;
      this.text += '!';
    }
  }
})
```

Contenido

Atributo

Evento

```
Mycomponent.vue

<template>
  <h2>Este es mi componente</h2>
  <p>{{number}}</p>
  <p :title="text"></p>
  <button @click="buttonClicked">Haz click</button>
</template>

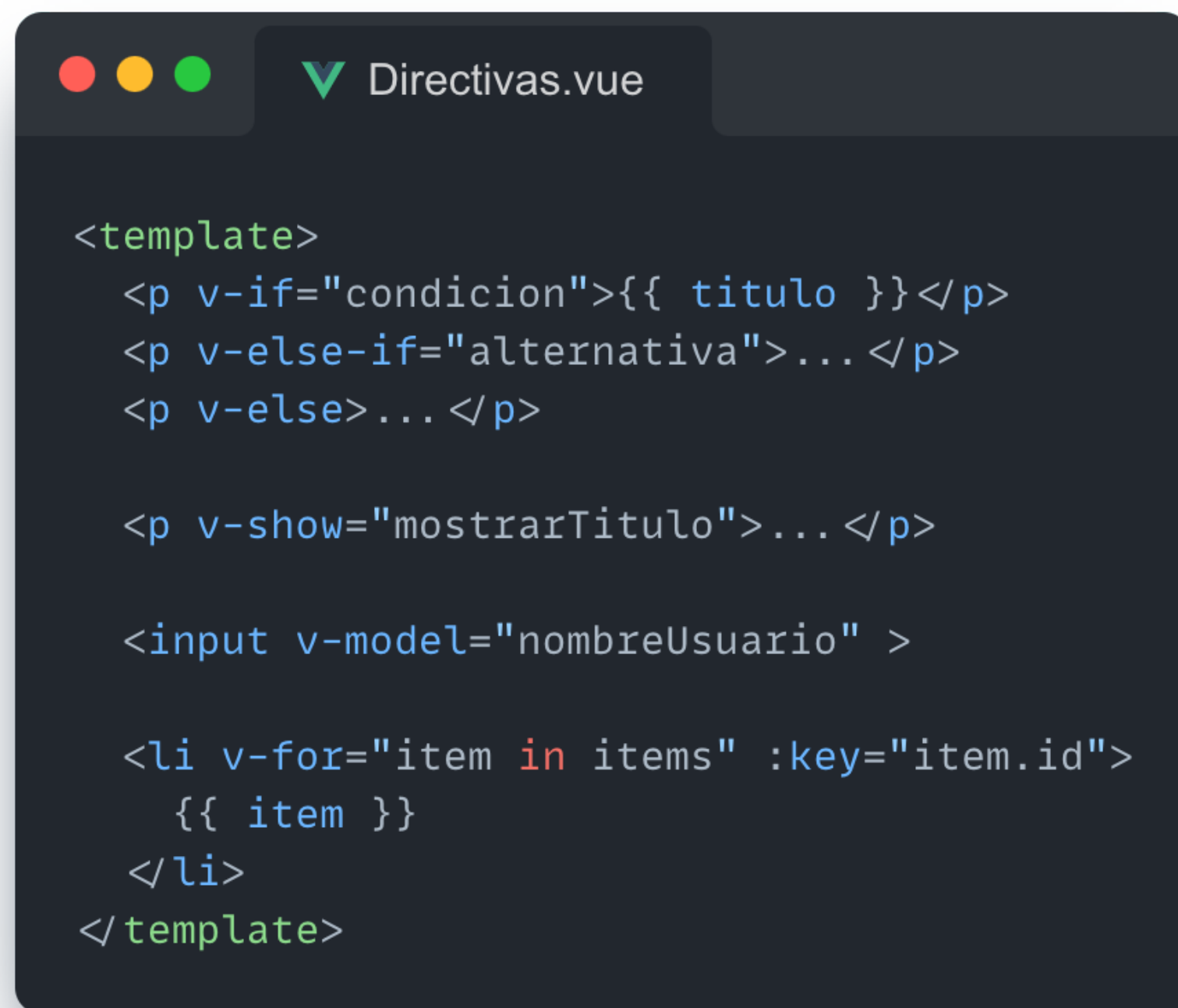
<script lang="ts">
import { defineComponent } from 'vue'

export default defineComponent({
  name: 'MyComponent',
  data(){
    return{
      number:0,
      text: 'Hola'
    }
  },
  methods:{
    buttonClicked(){
      this.number++;
      this.text += '!';
    }
  }
})
```



# 2. Interacción con el DOM

## Directivas



```
<template>
  <p v-if="condicion">{{ titulo }}</p>
  <p v-else-if="alternativa">...</p>
  <p v-else>...</p>

  <p v-show="mostrarTitulo">...</p>

  <input v-model="nombreUsuario" >

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



# 2. Interacción con el DOM

## Eventos

A screenshot of a code editor window titled 'Eventos.vue'. The editor shows Vue.js template syntax for event binding. It includes three buttons with different event binding methods: 'v-on:click', '@click', and '@click' with a function argument. It also shows a form with a submit button and an input field, both using event binding. The code is as follows:

```
<template>
  <button v-on:click="comprar">+</button>
  <button @click="comprar">+</button>
  <button @click="comprar(producto)">+</button>

  <form @submit.prevent="addProduct">
    <button type="submit" @click.once="comprar">+</button>
    <input @keyup.enter="enviar">
  </form>
</template>
```

Modificadores de eventos:

.stop .prevent .self .capture .once .passive .exact

Modificadores de tecla:

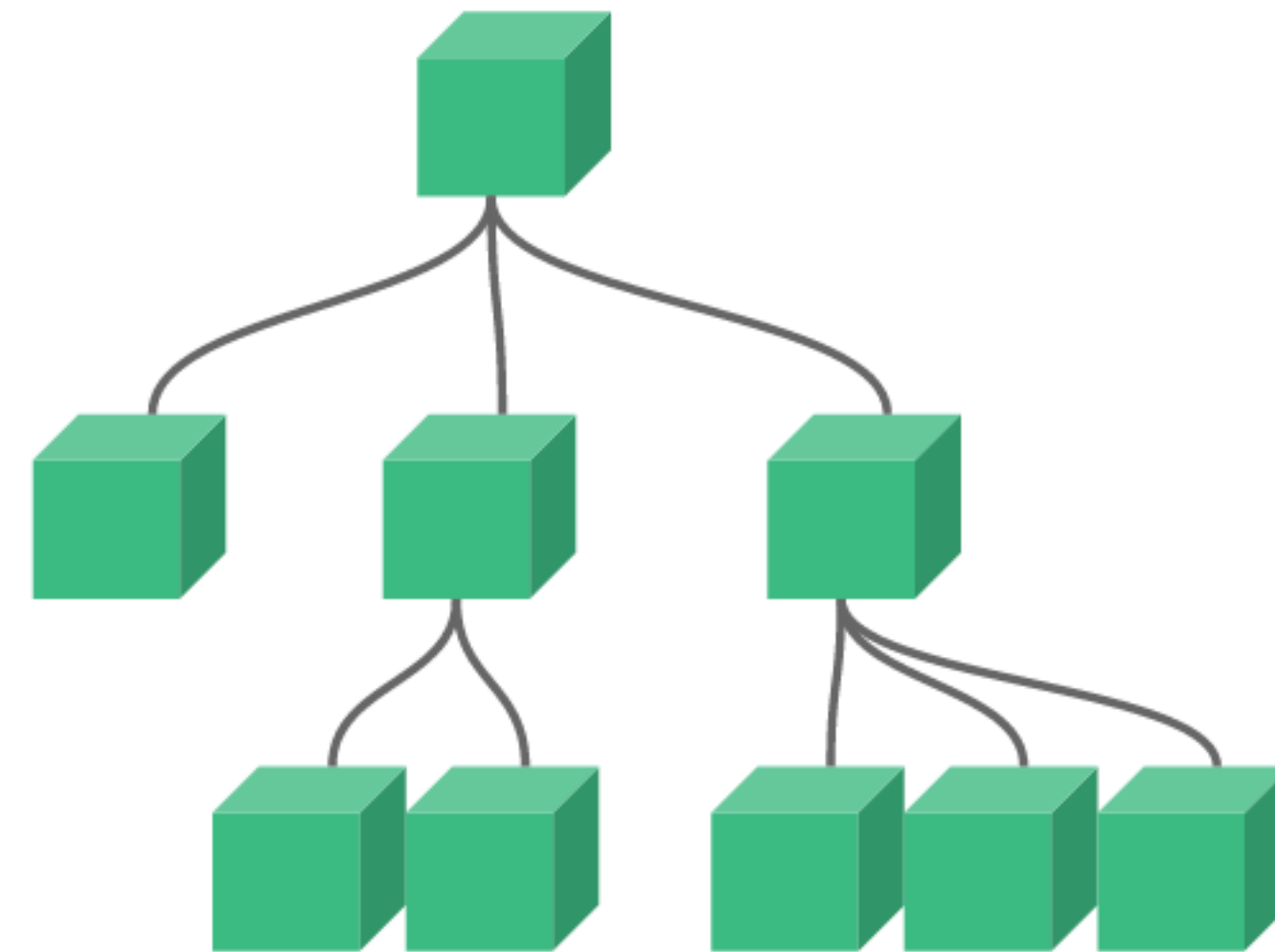
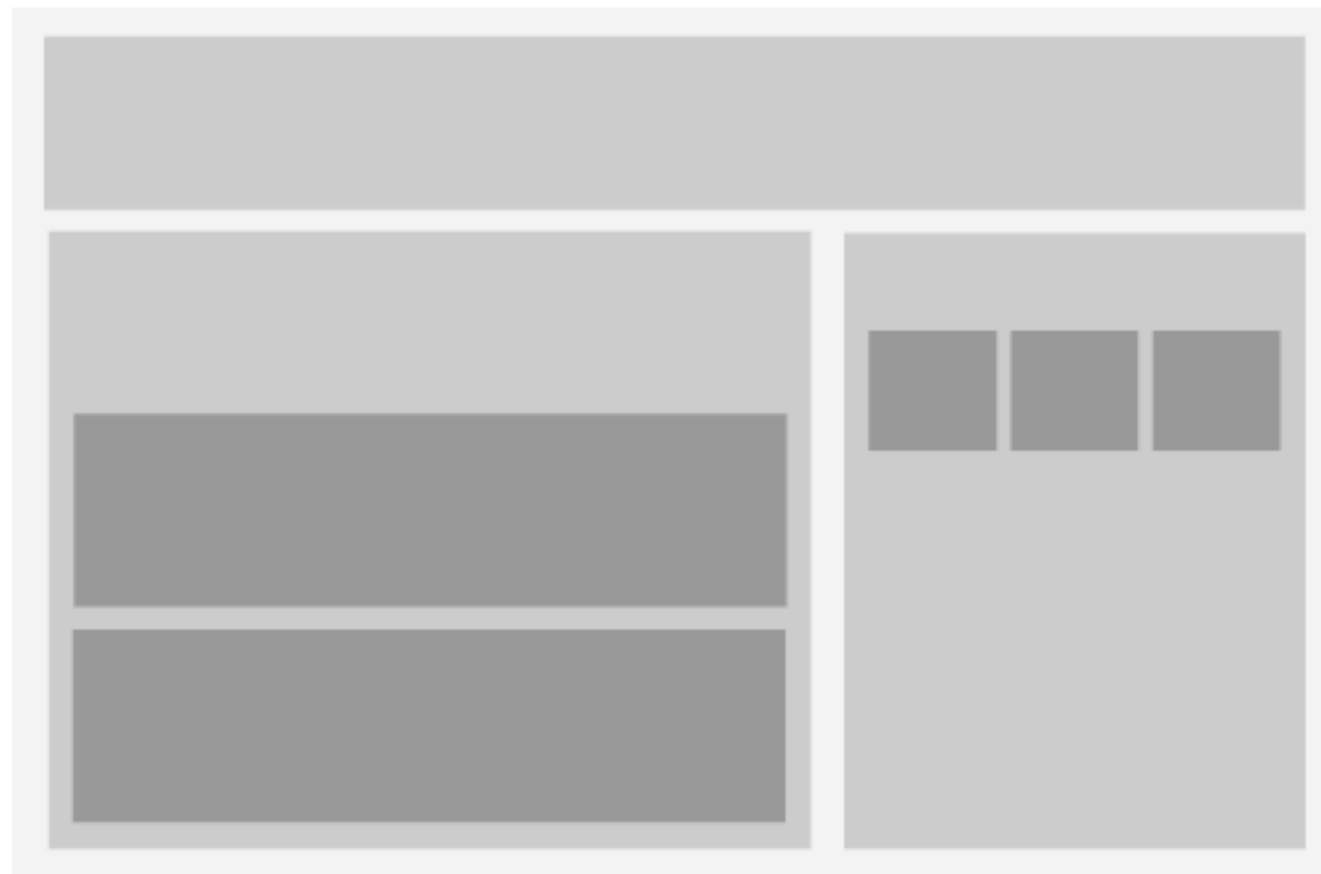
.enter .tab .delete .esc .space .up .down .left .right .ctrl .alt .shift .meta

Modificadores de ratón:

.left .right .middle

# 3. Componentes

Introducción y conceptos básicos



# 3. Componentes

## Estructura

```
ComponentStructure.vue

<script lang="ts">
import { defineComponent } from 'vue'

export default defineComponent({
  components: { ... },
  props: {
    titular: String,
  },
  data: function() {
    return {
      nombre: 'Juan',
      edad: 45
    }
  },
  computed: {
    datosUsuario: function () {
      return this.nombre + ': ' + this.edad
    }
  },
  watch: { ... },
  methods: { ... },
  setup: { ... }
})
</script>
```

Ciclo de vida de un componente:

- beforeCreate
- created
- beforeMount
- mounted
- beforeUpdate
- updated
- beforeDestroy
- destroyed

# 3. Componentes

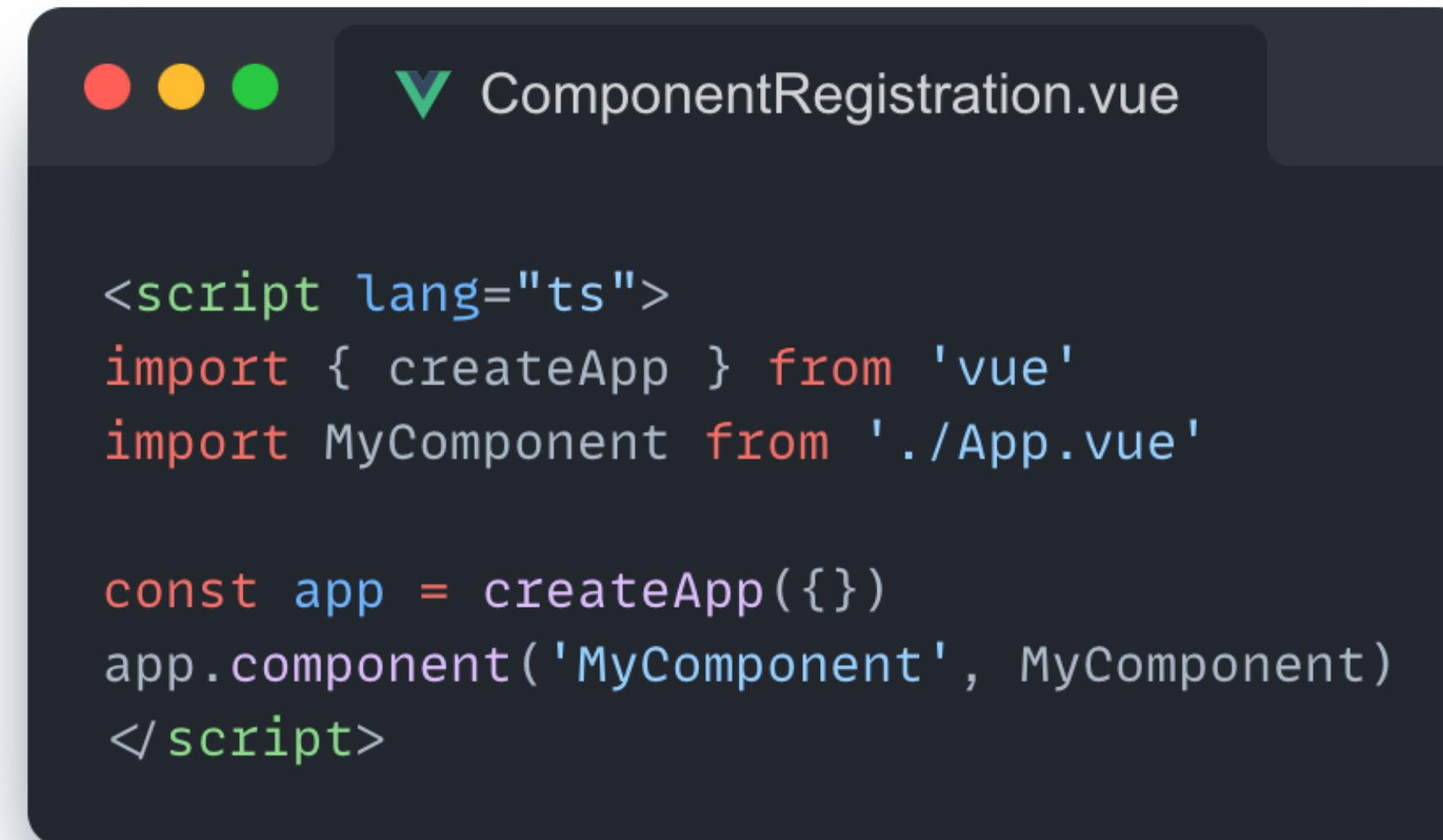
## Propiedades disponibles

Instance props:

- **\$data**
- **\$props**
- \$el
- \$options
- \$parent
- \$root
- \$slots
- **\$refs**
- \$attrs
- \$watch()
- **\$emit()**
- \$forceUpdate()
- **\$nextTick()**

# 3. Componentes

Registro global o local de componentes



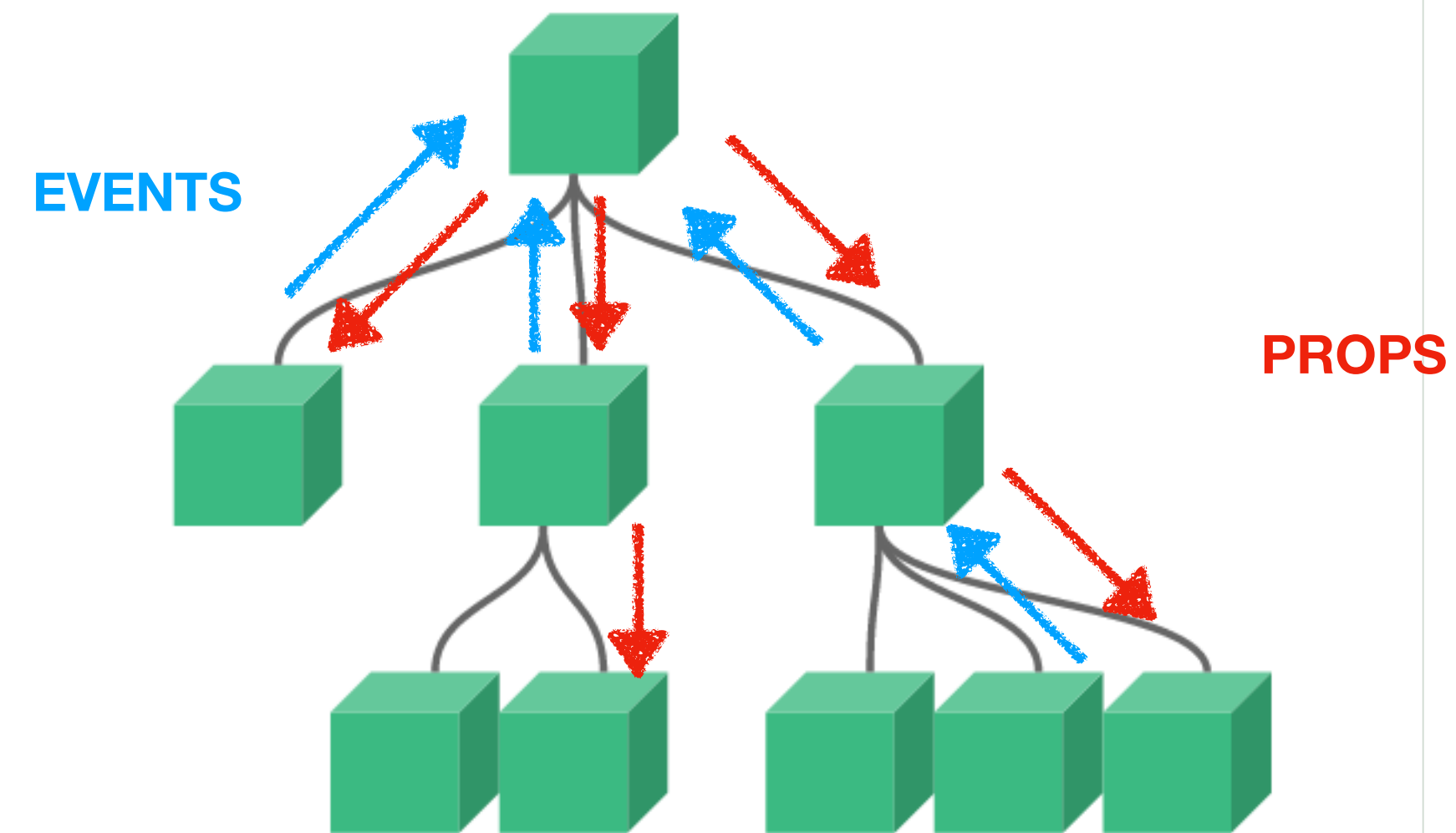
```
<script lang="ts">
import { createApp } from 'vue'
import MyComponent from './App.vue'

const app = createApp({})
app.component('MyComponent', MyComponent)
</script>
```

\*No recomendado (tree-shaking y redundancia)

# 4. Comunicación entre componentes

## Paradigma de vue



\*Este paradigma aporta seguridad y eficiencia, pero produce situaciones poco manejables en apps complejas.

# 4. Comunicación entre componentes

## Validación de props



```
ComponentRegistration.vue

<script lang="ts">
import { defineComponent } from 'vue'

export default defineComponent({
  components: { ... },
  props: {
    titular: String,
    email: {
      type: String,
      required: true,
      default: 'none'
      validator: function (value) {
      }
    }
  },
  data: function() {
    return {
      nombre: 'Juan',
      edad: 45
    }
  }
})
</script>
```



# 4. Comunicación entre componentes

## Events

```
ParentComponent.vue

<template>
  <ChildComponent @incremento="(n) => total += n" />
</template>
```



```
ChildComponent.vue

<template>
  <button @click="$emit('incremento', 1)">
    Sumar 1
  </button>
</template>
```

# 5. Uso avanzado de componentes

## Composable functions

```
mouse.ts

import { ref, onMounted, onUnmounted } from 'vue';

export function useMouse() {
  const x = ref(0);
  const y = ref(0);

  function update(event: MouseEvent) {
    x.value = event.clientX;
    y.value = event.clientY;
  }

  onMounted(() => window.addEventListener('mousemove', update));
  onUnmounted(() => window.removeEventListener('mousemove', update));

  return { x, y };
}
```



```
MyComponent.vue

<script lang="ts">
import { defineComponent } from 'vue';
import { useMouse } from '../mouse';

export default defineComponent({
  name: "MyComponent",
  data(){ ... },
  setup(){
    const mouse = useMouse();
    return {
      mouse
    }
  }
});
</script>
```

# 6. Directivas

## Directivas disponibles

Directivas Vue.js 3:

- v-text
- v-html
- v-show
- v-if
- v-else
- v-else-if
- v-for
- v-on
- v-bind
- v-model
- v-slot:slotname
- v-pre
- v-once

# 6. Directivas avanzadas

## Directivas propias

```
CustomDirective.vue

<script lang="ts">
import { defineComponent } from 'vue';
import { useMouse } from '../mouse';

const focus = {
  mounted: (el: any) => el.focus()
}

export default defineComponent({
  ...,
  directives: {
    focus
  }
});
</script>
```

```
CustomDirective.vue

<template>
  <input v-focus />
</template>
```

# Sesión 1

## Recursos

<https://vuejs.org/guide/introduction.html>

<https://cli.vuejs.org/>

<https://www.vuemastery.com/pdf/Vue-Essentials-Cheat-Sheet.pdf>