条件渲染

<template> 上的 v-if

因为 v-if 是一个指令，他必须依附于某个元素。但如果我们想要切换不止一个元素呢？在这种情况下我们可以在一个 <template> 元素上使用 v-if，这只是一个不可见的包装器元素，最后渲染的结果并不会包含这个 <template> 元素。

```
<template v-if="ok">
  <h1>Title</h1>
  <p>Paragraph 1</p>
  <p>Paragraph 2</p>
</template>
```

v-else 和 v-else-if 也可以在 <template> 上使用。

v-show

另一个可以用来按条件显示一个元素的指令是 v-show。其用法和 v-if 基本一样：

```
<h1 v-show="ok">Hello!</h1>
```

不同之处在于 v-show 会在 DOM 渲染中保留该元素；v-show 仅切换了该元素上名为 display 的 CSS 属性。

v-show 不支持在 <template> 元素上使用，也不能和 v-else 搭配使用。

v-if 有更高的切换开销，而 v-show 有更高的初始渲染开销。因此，如果需要频繁切换，则使用 v-show 较好；如果在运行时绑定条件很少改变，则 v-if 会更合适。

v-if 和 v-for

Vue2 和 Vue3 在官方文档中，都不推荐 v-for 和 v-if 同时使用。

Vue2 中 v-for 比 v-if 优先级更高，先循环再判断，满足 if 条件的才渲染，不满足的就不渲染，就会造成多余的循环，影响性能。

Vue3 中 v-if 比 v-for 优先级高，先判断，满足 if 条件的再循环、渲染，不满足的就不渲染，不会影响性能。但是代码会报错，因为 Vue3 底层会转换代码：

```
<template>
  <div v-for="item if [1, 2, 3]" v-if="item !== 2"></div>
</template>
```

转换成：

```
<template>
  <template v-if="item !== 2">
    <div v-for="item if [1, 2, 3]"></div>
  </template>
</template>
```