v-show

写法：v-show='表达式'

适用于：切换频率较高的场景

特点：不展示的 DOM 元素不被移除，仅仅是通过样式进行隐藏。就是添加了一个`display:none`属性

```
    <div id="a">
        <h2 v-show="a">文字显示与隐藏</h2>
        <h3 v-show="1 === 2">文字显示与隐藏</h3>
    </div>

    <script>
        const vm = new Vue({
            el:"#a",
            data:{
                a:false
            }
        })
    </script>
```

v-if

写法：(1) v-if='表达式'

(2) v-else-if='表达式'

(3) v-else='表达式'

适用于：切换频率较低的场景，避免对页面进行重绘

特点：

不展示的 DOM 元素，直接将**元素移除**

注意：v-if可以和 v-else-if、v-else一起使用，但要求结构不能被打断。

v-if='表达式'

```
<div id="a">
    <h2 v-if="a">文字显示与隐藏</h2>
    <h3 v-if="1 !=== 1">欢迎</h3>
</div>

<script>
    const vm = new Vue({
        el:"#a",
        data:{
            a:false
        }
    })
</script>
```

例子：例子,使用频率高的用v-show，频率低的v-if，因为v-if会重新渲染元素

1. 不允许被其他元素打断

```
<div id="a">
    <h2>当前的n值时{{n}}</h2>
    <button @click="n++">点击</button>
    <div v-if="n===1">1</div>
    <div v-else-if="n===2">2</div>
    <div v-else>3</div>
</div>

<script>
    const vm = new Vue({
        el:"#a",
        data:{
            n:0
        }
    })
</script>
```

`<template>` 模板

`v-if`与`<template>`配合使用

```
<div id="a">
    <template v-if="true">
        <h2>你好</h2>
        <h2>UF</h2>
        <h2>西安</h2>
    </template>
</div>

<script>
    const vm = new Vue({
        el:"#a",
        data:{
```

```
                n:0
            }
        })
    </script>
```