第5章 样式绑定

操作页面元素的 class 属性和 style 属性是数据绑定的一个常见的需求，开发者可以使用 v-bind 来处理 class 属性和 style 属性。由于采用字符串拼接为属性赋值比较麻烦且容易出错，因此在 Vue 3 中可以使用对象或数组为属性赋值。

5.1 绑定 HTML 样式

给页面元素设置 class 属性可以更改元素的样式，class 属性值指定样式表的类选择器。

5.1.1 对象控制样式

开发者可以通过传给 v-bind：class 一个对象，从而动态地切换 class 属性值，代码如下：

```
<div v-bind:class="{ active: isActive }"></div>
```

上述代码表示 active 这个样式存在与否取决于 isActive 的值，可以通过以下代码来查看 isActive 值的改变对于样式的影响：

```
<div id='app'>
 <div v-bind:class="{active:isActive}"></div>
</div>

<script>
 const app = {
  data() {
   return {
    isActive: true
   };
  }
 };
 const vm = Vue.createApp(app).mount('#app');
</script>
```

渲染结果如图 5.1 所示。

在控制台中输入 vm. $data.isActive = false，可以看到页面发生改变，如图 5.2 所示。

当 isActive 值为 true 时，active 样式起作用，开发者可以在对象中传入更多字段来动态切换多个样式。此外，v-bind：class 指令也可以与普通的 class 属性共存，尝试修改如下模

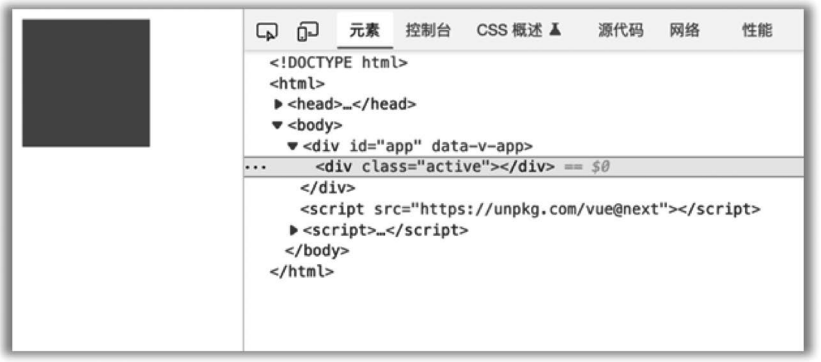

图 5.1　class 属性绑定对象的渲染结果

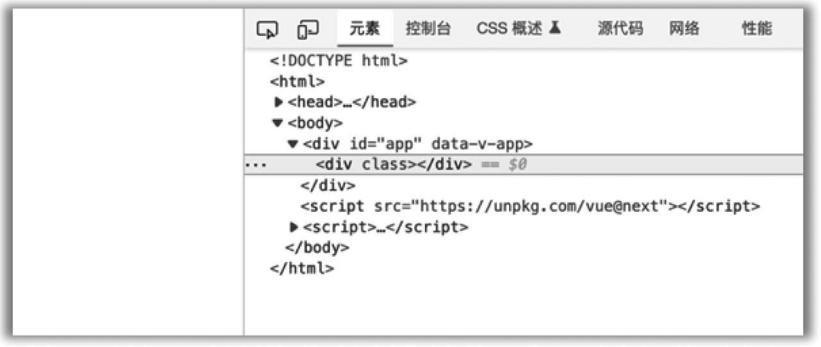

图 5.2　isActive 值改变为 false 后的页面变化

板和 data 代码：

```
<div
 class="static"
 v-bind:class="{ active: isActive, 'text-danger': hasError }"
></div>
data: {
   isActive: true,
   hasError: false
  }
```

渲染结果如图 5.3 所示。

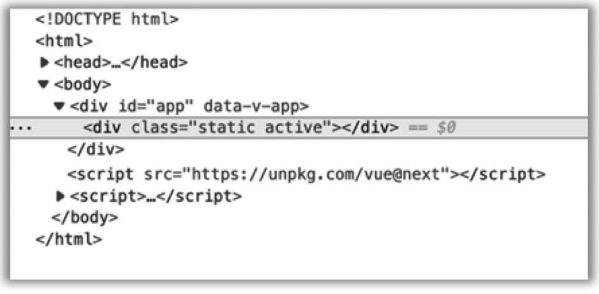

图 5.3　v-bind：class 指令与 class 属性共存的渲染结果

当 isActive 或者 hasError 发生变化时，class 属性的样式也会相应地更新，如将 hasError 的值设置为 true，则 class 属性的样式将变为“static active text-danger”。绑定的数据对象不必内联定义在模板中，如果绑定的数据对象比较复杂，可以在数据属性中单独定义一个对象进行绑定，代码如下：

```
<div id = "app">
 <div v - bind:class = "classObject "></div>
</div>
<script>
 const vm = Vue.createApp(()
 data() {
  return{
   classObject: {
    active: true,
    'text - danger': false
   }
  }
 }
 }) .mount('#app');
</script>
```

使用计算属性也可以实现动态更改样式，可以使用一个返回对象的计算属性进行绑定，代码如下：

```
<div v - bind:class = "classObject"></div>
<script>
 const vm = Vue.createApp({
  data() {
   return {
    isActive: true,
    error: null
   }
  },
  computed: {
   classObject() {
    return {
     active: this.isActive && !this.error,
     'text - danger': this.error && this.error.type === 'fatal'
    }
   }
  }
 }).mount('#app');
</script>
```

5.1.2　数组控制样式

开发者可以将数组传给 v-bind：class，这种方式将会在元素上应用一个样式列表，代码如下：

```
<style>
 .active{
```

```
    width:100px;
    height:100px;
    background:green;
   }
   .text-danger {
    background:red;
   }
</style>

<div v-bind:class="[activeClass, errorClass]"></div>
<script>
 const vm = Vue.createApp({
  data() {
   return {
    activeClass: 'active',
    errorClass: 'text-danger'
   }
  }
 }).mount('#app');
</script>
```

如果开发者想根据条件切换样式列表中的样式，可以使用三元表达式，代码如下：

```
<div v-bind:class="[isActive?activeClass:'',errorClass]"></div>
<script>
 const vm = Vue.createApp({
  data() {
   return {
    activeClass: 'active',
    errorClass: 'text-danger',
    isActive: true
   }
  }
 }).mount('#app');
</script>
```

当有多个条件时上述写法有些烦琐，在数组语法中可以使用简化的对象语法，代码如下：

```
<div v-bind:class="[{ active: isActive }, errorClass]"></div>
```

5.1.3 在组件中的应用

在一个自定义单根元素组件上使用 class 属性时，会把相应样式添加到该组件的根元素上，而不会覆盖根元素上已有的样式，示例代码如下：

```
Vue.component('my-component', {
 template: '<p class="foo bar">Hi</p>'
})
<my-component class="baz boo"></my-component>
<p class="foo bar baz boo">Hi</p>
```

对于带数据绑定的 class 属性同样适用，代码如下：

```
<my-component v-bind:class="{ active: isActive }"></my-component>
<p class="foo bar active">Hi</p>
```

如果组件有多个根元素，则需要通过使用 $attrs 组件属性来指定哪个根元素接收这个 class 属性，代码如下：

```
<div id="app">
 <my-component class="baz"></my-component>
</div>
<script>
 const app = Vue.createApp({})
 app.component('my-component', {
  template: '
    <p :class="$attrs.class">Hi!</p>
    <span>This is a child component</span>
   '
 })
</script>
```

5.2 绑定内联样式

使用 v-bind：style 可以给元素绑定内联样式。

5.2.1 对象描述样式

v-bind：style 的对象语法非常直观，看起来非常像 CSS，但实际上是一个 JavaScript 对象。CSS 属性名可以使用驼峰式（camelCase）或短横线分隔（kebab-case）来命名，代码如下：

```
<div id="app">
 <div v-bind:style="{ color: activeColor, fontSize: fontSize + 'px' }"></div>
</div>
<script>
 const vm = Vue.createApp({
  data() {
   return {
    activeColor: 'red',
    fontSize: 30
   }
  }
 }).mount('#app');
</script>
```

如果直接使用对象字符串的方式设置 CSS 样式属性，那么代码冗长且可读性较差，可以在数据属性中定义一个样式对象，通过 v-bind：style 绑定该对象让模板更清晰，代码如下：

```
<div id="app">
 <div v-bind:style="styleObject"></div>
</div>
```

```
<script>
 const vm = Vue.createApp({
  data() {
   return {
    styleObject: {
     color: 'red',
     fontSize: '13px'
    }
   }
  }
 }).mount('#app');
</script>
```

5.2.2 数组描述样式

v-bind：style 的数组语法可以将多个样式对象应用到同一个元素上，代码如下：

```
<div id="app">
 <div v-bind:style="[baseStyles,moreStyles]"></div>
</div>
<script>
 const vm = Vue.createApp({
  data() {
   return {
    baseStyles: {
     border: 'solid 2px black'
    },
    moreStyles: {
     width: '100px',
     height: '100px',
     background: 'orange',
    }
   }
  }
 }).mount('#app');
</script>
```

5.2.3 自动添加前缀

当 v-bind：style 使用需要添加浏览器引擎前缀的 CSS 属性时，Vue 3 会自动侦测并添加相应的前缀，结果如下所示：

```
transform:rotate(7deg);
-ms-transform:rotate(7deg);                    // IE 9
-moz-transform:rotate(7deg);                   //Firefox
-webkit-transform:rotate(7deg);                //Safari 和 Chrome
-o-transform:rotate(7deg);                     // Opera
```

5.2.4 多重值样式

可以为绑定的 style 属性赋一个包含多个带前缀值的数组，这样写只会渲染数组中最

后一个被浏览器支持的值。对于以下代码：

```
<div :style="{display:['-webkit-box','-ms-flexbox','flex']}"></div>
```

如果浏览器支持不带浏览器前缀的 flexbox，那么就只会渲染 display：flex。


视频讲解

5.3　实例：实现列表的奇偶行不同样式

在工程项目中经常需要使用表格来展示多行数据。当表格行数较多时，为了让用户能够区分不同的行，通常会针对奇偶行应用不同的样式，这样用户可以更加清晰地查看数据。

本例定义了一个针对偶数行的样式规则，代码如下：

```
.even {
 background-color: #cdedcd;
}
```

表格数据采用 v-for 循环输出，v-for 可以带一个索引参数，根据这个索引参数判断奇偶行。循环索引从 0 开始，对应的是第 1 行，为了方便判断，将其加 1 后再进行判断。判断规则为(index+1)%2===0，使用 v-bind：class 的对象语法，当该表达式值为 true 时，应用样式 even，完整代码如下所示：

```
<div id="app" v-cloak>
 <table>
  <tr>
   <th>序号</th>
   <th>课程</th>
   <th>教师</th>
   <th>课时</th>
   <th>操作</th>
  </tr>
  <tr v-for="(book, index) in books" :key="book.id" :class="{even:(index+1)%2===0}">
   <td>{{book.id}}</td>
   <td>{{book.title}}</td>
   <td>{{book.teacher}}</td>
   <td>{{book.classHour}}</td>
   <td>
    <button @click="deleteItem(index)">删除</button>
   </td>
  </tr>
 </table>
</div>

<script>
 const vm = Vue.createApp({
  data() {
   return {
    books: [
     {
      id: 1,
      title: '高等数学',
      teacher: '王老师',
      classHour: 32
```

```
        },
        {
         id: 2,
         title: 'VC++',
         teacher: '李老师',
         classHour: 8
        },
        {
         id: 3,
         title: '英语',
         teacher: '孙老师',
         classHour: 16
        },
        {
         id: 4,
         title: 'Web开发基础',
         teacher: '王老师',
         classHour: 16
        }
       ]
      }
     },
     methods: {
      deleteItem(index) {
       this.books.splice(index, 1);
      }
     }
    }).mount('#app');
</script>
```

上述代码在浏览器中的渲染结果如图 5.4 所示。

序号	课程	老师	课时	操作
1	高等数学	王老师	32	删除
2	VC++	李老师	8	删除
3	英语	孙老师	16	删除
4	Web开发基础	王老师	16	删除

图 5.4　列表的奇偶行不同的渲染结果

5.4　本章小结

本章介绍了 class 属性与 style 属性的绑定，开发者在使用 CSS 样式时可以拥有更多的选择和更灵活的处理方式。

习题

1. Vue 3 中绑定样式的方式有哪几种？区别是什么？
2. 使用样式实现一个 loading 效果。