第5章

绑定CSS样式

视频讲解

CSS 样式用来决定网页的外观。DOM 元素的 class 属性和 style 属性都用于设定 CSS 样式。例如,以下 元素的 class 属性设定字体为红色:

```
<style>
  .redtext {color: red}
</style>

<div class="redtext">Hello</div>
```

以下 元素的 style 属性设定字体为红色:

```
<div style="color: red">Hello</div>
```

以上 元素的 class 属性和 style 属性的取值是固定的。如果希望设置动态值,那么可以通过 Vue 的 v-bind 指令为 class 属性或 style 属性绑定一个动态值。本章介绍为 DOM 元素动态绑定 CSS 样式的方法。

5.1 绑定 class 属性

DOM 元素的 class 属性用来设定一个样式类型。在例程 5-1 中,定义了 static、redtext 和 bluetext 3 个样式类型。 元素的 class 属性包含了 static 和 redtext 这两个样式类型。

例程 5-1 original. html

```
<style>
  .static{
```

```
      width: 100px;
      height: 50px;
      text-align: center;
      background: yellow
    }
   .redtext {color: red}
   .bluetext{color: blue}
</style>

<div id="app">
  <div class="static redtext">Hello</div>
</div>
```

通过浏览器访问 original. html,会看到网页上有一个黄色的矩形框,矩形框中有一个红色的字符串 Hello。static 样式类型设定了 元素的大小和背景色,redtext 样式类型设定了字符串 Hello 的颜色。

如果希望能动态改变网页上字符串 Hello 的颜色,可以用 v-bind 指令把 元素的 class 属性与一个对象绑定。例如,以下 class 属性的值是一个对象{redtext: true}:

```
<div v-bind:class="{redtext: true}">Hello</div>
<!-- 简写为:-->
<div :class="{redtext: true}">Hello</div>
```

v-bind 指令对 元素的渲染结果为:

```
<div class="redtext">Hello</div>
```

以下 class 属性的值为一个对象{redtext: false}:

```
<div :class="{redtext: false}">Hello</div>
```

v-bind 指令对 元素的渲染结果为:

```
<div>Hello</div>
```

由此可见,v-bind 指令可以动态控制是否使用特定的样式类型。在例程 5-2 中,v-bind 指令把 class 属性与一个对象{redtext: isActive}绑定。isActive 是一个变量,网页上的“切换字体颜色”按钮使 isActive 变量的值在 true 和 false 之间切换。

例程 5-2　redtext. html

```
<div id="app">
  <div class="static" :class="{ redtext: isActive }">Hello</div>
  <button v-on:click="isActive=! isActive">切换字体颜色</button>
</div>

<script>
```

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

通过浏览器访问 redtext. html，会得到如图 5-1 所示的网页。

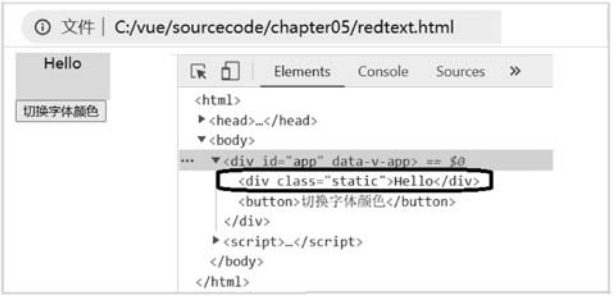

图 5-1　redtext. html 的网页

一开始，isActive 变量的值为初始值 false。从图 5-1 可以看出，v-bind 指令对 元素的渲染结果为：

```
<div class="static">Hello</div>
```

在 redtext. html 的网页上单击“切换字体颜色”按钮，isActive 变量的值变为 true，这时网页上 Hello 字符串变成红色。v-bind 指令对 元素的渲染结果为：

```
<div class="static redtext">Hello</div>
```

在例程 5-3 中，v-bind 指令为 元素的 class 属性绑定的对象为｛ redtext：isActive，bluetext：!isActive ｝。当 isActive 变量的值为 true， 元素的 class 属性的渲染结果为 static redtext；当 isActive 变量的值为 false， 元素的 class 属性的渲染结果为 static bluetext。

例程 5-3　redbluetext. html

```
<div id="app">
  <div class="static"
       :class="{ redtext: isActive,bluetext: ! isActive }">
    Hello
  </div>
  <button v-on:click="isActive=! isActive">切换字体颜色</button>
</div>

<script>
  const vm=Vue.createApp({
```

```
    data(){
      return {
        isActive: false
      }
    }
  }).mount('#app')
</script>
```

在 redbluetext. html 的网页上多次单击“切换字体颜色”按钮,会看到网页上的字符串 Hello 的颜色在红色与蓝色之间多次切换。

5.1.1　绑定对象类型的变量

如果与 DOM 元素的 class 属性绑定的对象很复杂,会影响组件的模板代码的可读性。为了简化模板代码,可以在组件的 data 选项中定义一个对象类型的变量,再把 DOM 元素的 class 属性与这个变量绑定。在例程 5-4 中, 元素的 class 属性与 textColor 变量绑定。

例程 5-4　classobject. html

```
<div id="app">
  <div class="static" :class="textColor">Hello</div>
</div>

<script>
  const vm=Vue.createApp({
    data(){
      return {
        textColor: { redtext: true, bluetext: false }
      }
    }
  }).mount('#app')
</script>
```

只要改变 textColor 变量,就能改变模板中<div class="static" : class="textColor" >元素的样式。

5.1.2　绑定计算属性

如果把 DOM 元素的 class 属性与一个计算属性绑定,就可以在计算属性的 get 函数中执行一些逻辑运算,从而动态改变 class 属性的取值。在例程 5-5 中, 元素的 class 属性与 textColor 计算属性绑定。

例程 5-5　classcomputed. html

```
<div id="app">
  <div class="static" :class="textColor">Hello</div>
```

```
    <button v-on:click="isActive=! isActive">
      切换字体颜色
    </button>
  </div>

  <script>
    const vm=Vue.createApp({
      data(){
        return {
          isActive: false
        }
      },
      computed:{          //计算属性
        textColor(){
          return {redtext:this.isActive,bluetext:! this.isActive }
        }
      }
    }).mount('#app')
  </script>
```

在 classcomputed. html 的网页上多次单击“切换字体颜色”按钮，会看到网页上的字符串 Hello 的颜色在红色与蓝色之间多次切换。

5.1.3 绑定数组

还可以把 DOM 元素的 class 属性与一个数组绑定，从而把多个样式类型赋值给 class 属性。在例程 5-6 中， 元素的 class 属性与数组[staticClass，redClass]绑定，这样， 元素就同时具有 static 样式类型和 redtext 样式类型。

例程 5-6　classarray. html

```
  <div id="app">
    <div :class="[ staticClass,redClass ]">Hello</div>
  </div>

  <script>
    const vm=Vue.createApp({
      data(){
        return {
          staticClass: 'static',
          redClass: 'redtext'
        }
      }
    }).mount('#app')
  </script>
```

在例程 5-7 中，通过“?:”三元表达式决定是否使用 redClass 变量指定的样式。

例程 5-7　dynamicarray. html

```
<div id="app">
  <div :class="[staticClass, isActive ?redClass : '']">Hello</div>
  <!--
  <div :class="[staticClass, {redtext:isActive} ]">Hello</div>
  -->
  <button v-on:click="isActive=! isActive">切换字体颜色</button>
</div>

<script>
  const vm=Vue.createApp({
    data(){
      return {
        isActive: false,
        staticClass: 'static',
        redClass: 'redtext'
      }
    }
  }).mount('#app')
</script>
```

如果与 class 属性绑定的数组中包含多个三元表达式,会使得代码比较臃肿,为了简化代码,可以用 5. 1. 1 节介绍的{样式类型名: true|false}对象替代三元表达式。例如,可以把 dynamicarray. html 中的 元素改写为:

```
<div :class="[staticClass, {redtext:isActive}]">Hello</div>
```

当 isActive 变量为 true,class 属性就会包含 redtext 样式类型; 当 isActive 变量为 false, class 属性就不会包含 redtext 样式类型。

5. 1. 4　为 Vue 组件绑定 CSS 样式

Vue 组件也具有 class 属性。在例程 5-8 中,定义了一个 item 组件。在 item 组件的模板中, 元素的 class 属性的值为 static。

例程 5-8　component. html

```
<div id="app">
  <item class="redtext"></item>
  <item :class="{bluetext: isActive}"></item>
</div>

<script>
  const app=Vue.createApp({
    data(){
      return { isActive: true }
    }
```

```
  })

  app.component('item',{
    template:'<div class="static">Hello</div> '       //item 组件的模板
  })
  app.mount('#app')
</script>
```

在 component. html 的根组件的模板中，加入了两个 item 组件，它们都设置了 class 属性，如：

```
<div id="app">
  <item class="redtext"></item>
  <item :class="{bluetext: isActive}"></item>
</div>
```

对于第一个 item 组件，其模板的渲染结果为：

```
<div class="static redtext">Hello</div>
```

对于第二个 item 组件，通过 v-bind 指令把 class 属性与{bluetext：isActive}对象绑定，isActive 变量的值为 true。item 组件的模板的渲染结果为：

```
<div class="static bluetext">Hello</div>
```

5.2 绑定 style 属性

DOM 元素的 style 属性可以直接指定具体的 CSS 样式。例如在以下代码中， 元素的 style 属性指定网页上 Hello 字符串的颜色为红色，字体大小为 17px：

```
<div style="color: red ; fontSize: 17px ">Hello</div>
```

如果通过 v-bind 指令把 style 属性绑定到一个对象，就可以动态改变 style 属性的值。以下代码把 style 属性与对象{color：redColor，fontSize：size+'px' }绑定：

```
<div v-bind:style="{ color: redColor, fontSize: size+'px' }">
  Hello
</div>

<!-- 简写为 -->
<div :style="{ color: redColor,fontSize: size+'px' }">
  Hello
</div>
```

只要改变 size 变量的大小，就能改变网页上 Hello 字符串的字体大小。

在例程 5-9 中,size 变量的大小可通过单击“字体变大”按钮改变。

例程 5-9　stylebind. html

```
<div id="app">
  <div :style="{ color: redColor,fontSize: size+'px' }">
    Hello
  </div>
  <button v-on:click="size++">字体变大</button>
</div>

<script>
  const vm=Vue.createApp({
    data(){
      return {
        redColor: 'red',
        size:17
      }
    }
  }).mount('#app')
</script>
```

通过浏览器访问 stylebind. html,会得到如图 5-2 所示的网页。单击网页上的“字体变大”按钮,会看到网页上的 Hello 字符串的字体不断变大。

图 5-2　stylebind. html 的网页

5.2.1　绑定对象类型的变量

如果与 DOM 元素的 style 属性绑定的对象很复杂,会影响模板代码的可读性。为了简化模板代码,可以在 Vue 组件的 data 选项中定义一个对象类型的变量,再把 DOM 元素的 style 属性与这个变量绑定。在例程 5-10 中, 元素的 style 属性与 styleObject 变量绑定。

例程 5-10　styleobject. html

```
<div id="app">
  <div :style="styleObject">Hello</div>
</div>

<script>
  const vm=Vue.createApp({
    data(){
      return {
        styleObject: { color: 'red',fontSize: '17px' }
```

```
        }
      }
    }).mount('#app')
</script>
```

只要改变 styleObject 变量，就能改变模板中<div ：style="styleObject">元素的样式。

5.2.2 绑定数组

还可以把 DOM 元素的 style 属性与一个数组绑定，从而把多个样式赋值给 style 属性。在例程 5-11 中， 元素的 style 属性与数组[staticStyle，redStyle]绑定，这样， 元素就同时具有 staticStyle 变量和 redStyle 变量指定的样式。

例程 5-11 stylearray. html

```
<div id="app">
  <div :style="[staticStyle, redStyle]">Hello</div>
</div>

<script>
  const vm=Vue.createApp({
    data(){
      return {
        staticStyle: {
          width:'100px',
          height:'50px',
          textAlign:'center',
          background: 'yellow'
        },
        redStyle: { color: 'red' }
      }
    }
  }).mount('#app')
</script>
```

5.2.3 与浏览器兼容

由于每个浏览器对 CSS 样式的支持程度不一样，因此一个 HTML 文档的 CSS 样式可能不被所有的浏览器兼容。v-bind 指令会识别各个浏览器对 CSS 样式的支持细节，确保 HTML 文档的 CSS 样式能够被当前浏览器正确解析。

v-bind 指令为 CSS 样式的兼容主要做了以下两方面的工作。

1. 自动添加前缀

随着 CSS 样式不断升级，有些新出现的样式属性不被所有浏览器支持。例如，新出现的 transform 样式属性用于对 DOM 元素进行旋转、缩放和移动等。针对不同的浏览器，需要

为该属性添加相应的内核引擎前缀,从而使浏览器支持该属性。在浏览器 IE9 中,该属性添加前缀后变为-ms-transform; 在浏览器 Safari 和 Chrome 的一些版本中,该属性添加前缀后变为-webkit-transform。随着浏览器本身的升级换代,前缀的名称也可能会发生变化。v-bind 指令能够根据当前使用的浏览器自动为 transform 属性添加前缀,保证当前浏览器正确解析 transform 属性。

2. 自动选择合适的样式属性值

同一个样式属性在不同的浏览器中可能有不同的取值。v-bind 指令允许把一个数组赋值给样式属性,v-bind 指令会自动从数组中选择适合当前浏览器的属性值,把它赋值给样式属性。例程 5-12 演示了 v-bind 指令为了确保 CSS 样式与浏览器兼容所做的工作。

例程 5-12　styleprefix. html

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

<script>
  const vm=Vue.createApp({
    data(){
      return {
        styleObject: {
          transform: 'rotate(45deg)',
          width: '100px',
          height: '50px',
          margin: '50px',
          textAlign: 'center',
          background: 'yellow'
        }
      }
    }
  }).mount('#app')
</script>
```

在 styleprefix. html 中,以下 元素的 style 属性与 styleObject 变量绑定:

```
<div :style="styleObject">Hello</div>
```

在 styleObject 变量中设定了 transform 样式属性,代码如下:

```
transform: 'rotate(45deg)'          //旋转 45 度
```

v-bind 指令会依据当前浏览器的类型,自动为 transform 样式属性添加相应的前缀,确保当前浏览器能正确解析 transform 样式属性。

以下 元素的 style 属性中设定了 display 样式属性,它的值是一个数组:

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

v-bind 指令会从数组中选择被当前浏览器支持的最后一个元素。例如，假定当前浏览器支持 display 属性的值为-webkit-box 和 flex，那么以上 元素的渲染结果为：

```
<div style="display: flex;">
```

通过 Chrome 浏览器访问 styleprefix. html，会得到如图 5-3 所示的网页。

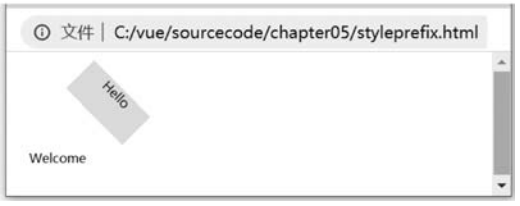

图 5-3　styleprefix. html 的网页

从图 5-3 可以看出，Chrome 浏览器会正确解析 styleprefix. html 的 CSS 样式。

5.3　范例：变换表格奇偶行的样式

对于 HTML 表格，为了制造醒目的视觉效果，可以用不同的颜色标识特定行，如用不同的颜色标识奇数行和偶数行。以下代码通过 v-bind 指令把<tr>元素的 class 属性与对象{markline : index % 2 !== 0}绑定：

```
<tr v-for="(book, index) in books"
            :class="{markline : index % 2 !== 0}">
  …
</tr>
```

当 index 变量为偶数，index % 2 !== 0 的值为 false，markline 样式不会被添加到 class 属性中；当 index 变量为奇数，index % 2 !==0 的值为 true，markline 样式会被添加到 class 属性中。表格中第一行的 index 变量值为 0。

例程 5-13 按照上述方式用不同的颜色标识表格中的奇数行与偶数行。

例程 5-13　books. html

```
<style>
  table {
    border: 1px solid black;
    width: 100%;
  }
  th { height: 50px; }
  th, td {
    border-bottom: 1px solid #ddd;
    text-align: center;
```

```
  }
  .markline {
    background-color: #FFCC66;
  }
</style>

<div id = "app">
  <table>
    <tr>
      <th>序号</th> <th>书名</th> <th>作者</th> <th>出版社</th>
    </tr>
    <tr v-for="(book, index) in books"
          :class="{markline : index %  2 !== 0}">
      <td>{{ index+1 }}</td>
      <td>{{ book.title }}</td>
      <td>{{ book.author }}</td>
      <td>{{ book.publisher }}</td>
    </tr>
  </table>
</div>

<script>
  const vm=Vue.createApp({
    data(){
      return {
        books:[
          { title: '《精通 Vue.js:Web 前端开发技术详解》',
            author: '孙卫琴',
            publisher: '清华大学出版社'},
          { title: '《精通 Spring:Java Web 开发技术详解》',
            author: '孙卫琴',
            publisher: '清华大学出版社'},
          …
        ]
      }
    }
  }).mount('#app')
</script>
```

通过浏览器访问 books. html,会得到如图 5-4 所示的网页。

序号	书名	作者	出版社
1	《精通Vue.js：Web前端开发技术详解》	孙卫琴	清华大学出版社
2	《基于Spring：Java Web开发技术详解》	孙卫琴	清华大学出版社
3	《精通JPA与Hibernate: Java对象持久化技术详解》	孙卫琴	清华大学出版社
4	《大话Java程序设计从入门到精通》	孙卫琴	电子工业出版社
5	《Tomcat与Java Web开发技术详解》	孙卫琴	电子工业出版社
6	《Java面向对象编程》	孙卫琴	电子工业出版社
7	《精通Struts: 基于MVC的Java Web设计与开发》	孙卫琴	电子工业出版社
8	《Java网络编程核心技术详解》	孙卫琴	电子工业出版社
9	《Java逍遥游记》	孙卫琴	电子工业出版社
10	《Hibernate逍遥游记》	孙卫琴	电子工业出版社
11	《Java 2认证考试指南与试题解析》	孙卫琴	上海科学技术出版社

图 5-4　books. html 的网页

5.4 小结

本章介绍了通过 v-bind 指令为 DOM 元素动态绑定 CSS 样式的方法。v-bind 指令可以把 DOM 元素的 class 属性或 style 属性与一个对象绑定，例如：

```
<div v-bind:class="{redtext: true}">Hello</div>
<!-- 简写为：-->
<div :class="{redtext: true}">Hello</div>

<div v-bind:style="{color: redColor}">Hello</div>
<!-- 简写为：-->
<div :style="{color: redColor}">
```

以上{redtext：true}和{color：redColor}都是对象。如果要绑定的对象比较复杂，为了简化模板代码，可以在 Vue 组件的 data 选项中定义一个对象类型的变量，然后把 DOM 元素的 class 属性或 style 属性与这个变量绑定。例如，以下代码把 class 属性和 style 属性分别和 textColor 变量和 styleObject 变量绑定：

```
<div :class="textColor" >Hello</div>
<div :style="styleObject">Hello</div>

//textColor 和 styleObject 变量在 Vue 组件的 data 选项中定义
data(){
  return {
    textColor: { redtext: true, bluetext: false },
    styleObject: { color: 'red', fontSize: '17px' }
  }
}
```

还可以把一个数组与 class 属性或 style 属性绑定，这样就能包含多个样式。例如，以下 class 属性和 style 属性都与数组绑定：

```
<div :class="[staticClass,redClass]">Hello</div>
<div :style="[staticStyle,redStyle]">Hello</div>
```

以上[staticClass，redClass]数组和[staticStyle，redStyle]数组中的 staticClass、redClass、staticStyle 以及 redStyle 都是在 Vue 组件的 data 选项中定义的变量。

5.5 思考题

1. 对于以下代码：

```
<div :class="{redtext: true}">Hello</div>
```

:class 使用了(　　)指令的缩写形式。(单选)

A. v-model　　B. v-bind　　C. v-on　　D. v-show

2. 对于以下代码:

```
<style>
  .static{
     text-align: center;
     background: yellow
   }
  .redtext {color: red}
</style>

<div class="static" :class="{redtext: isActive}">Hello</div>
```

当 isActive 变量为 true, 元素的 class 属性的渲染结果是(　　)。(单选)

A. static　　B. redtext

C. static redtext: true　　D. static redtext

3. textStyle 变量的定义如下:

```
<script>
  const vm=Vue.createApp({
    data(){
      return {
        textStyle: { color: 'red' }
      }
    }
  }).mount('#app')
</script>
```

以下选项(　　)会在网页上用红色字体显示字符串 Hello。(多选)

A. <div :style="textStyle">Hello

B. <div :style="{textStyle: true}">Hello

C. <div :style="[textStyle]">Hello

D. <div :class="{textStyle: true}">Hello

4. bluetext 样式的定义如下:

```
<style>
  .bluetext {color: blue}
</style>
```

以下选项(　　)会在网页上用蓝色字体显示字符串 Hello。(多选)

A. <div :style="bluetext">Hello

B. Hello

C. <div :class="[bluetext]">Hello

D. <div :class="{bluetext: true}">Hello

5. 对于以下代码：

```
<style>
  .redtext {color: red}
</style>

<div id="app">
  <li v-for="n in 4" :class="{redtext: n<=2}">{{n}}</li>
</div>
```

在网页上会看到(　　)。(多选)

A. 网页上输出数字 1、2、3、4

B. 所有的数字为黑色字体

C. 所有的数字为红色字体

D. 数字 1 和 2 为红色字体