Vue基础 – v-model表单

王红元 coderwhy

目录
content

1 v-model基本使用

2 v-model绑定原理

3 v-model绑定radio

4 v-model绑定checkbox

5 v-model绑定select

6 v-model的修饰符

综合案例

- 现在我们来做一个相对综合一点的练习：**书籍购物车**

	书籍名称	出版日期	价格	购买数量	操作
1	《算法导论》	2006-9	¥85	- 1 +	移除
2	《UNIX编程艺术》	2006-2	¥59	- 1 +	移除
3	《编程珠玑》	2008-10	¥39	- 1 +	移除
4	《代码大全》	2006-3	¥128	- 1 +	移除

总价: ¥311

- **案例说明：**
  - 1.在界面上以表格的形式，显示一些书籍的数据；
  - 2.在底部显示书籍的总价格；
  - 3.点击+或者-可以增加或减少书籍数量（如果为1，那么不能继续-）；
  - 4.点击移除按钮，可以将书籍移除（当所有的书籍移除完毕时，显示：购物车为空~）；

v-model的基本使用

- **表单提交**是开发中非常常见的功能，也是和用户交互的重要手段：
  - 比如用户在登录、注册时需要提交账号密码；
  - 比如用户在检索、创建、更新信息时，需要提交一些数据；
- 这些都要求我们可以在**代码逻辑中获取到用户提交的数据**，我们通常会使用**v-model指令**来完成：
  - v-model指令可以在表单 input、textarea以及select元素上创建双向数据绑定；
  - 它会根据控件类型自动选取正确的方法来更新元素；
  - 尽管有些神奇，但 v-model 本质上不过是语法糖，它负责监听用户的输入事件来更新数据，并在某种极端场景下进行一些特殊处理；

```
<template id="my-app">
  <input type="text" v-model="message">
  <h2>{{message}}</h2>
</template>
```

v-model的原理

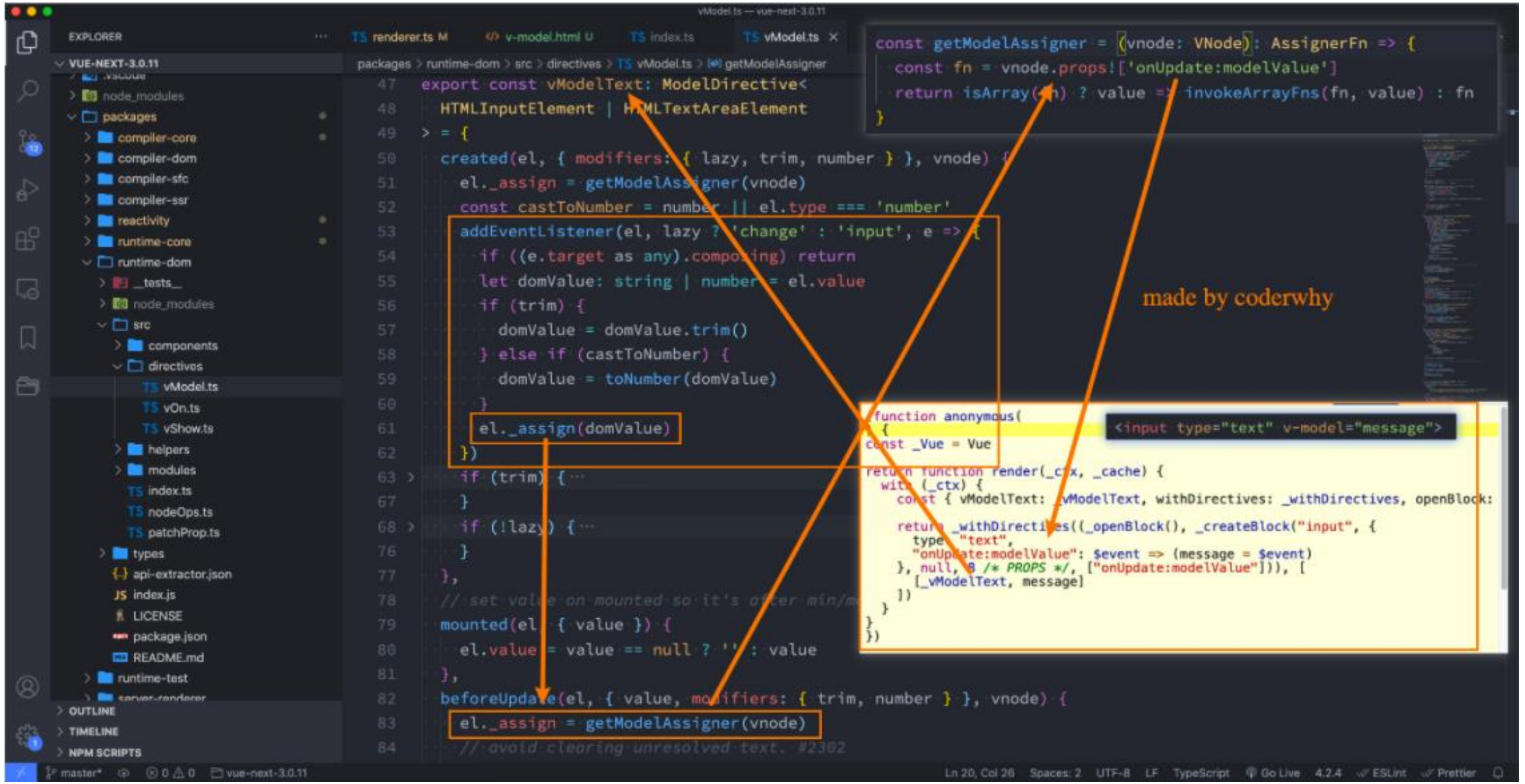
- **官方有说到，v-model的原理其实是背后有两个操作：**
  - v-bind绑定value属性的值；
  - v-on绑定input事件监听到函数中，函数会获取最新的值赋值到绑定的属性中；

```html
<input v-model="searchText" />
```

等价于：

```html
<input :value="searchText" @input="searchText = $event.target.value" />
```

事实上v-model更加复杂



v-model绑定textarea

- 我们再来绑定一下**其他的表单类型**：textarea、checkbox、radio、select

- **我们来看一下绑定textarea：**

```
<!-- 1.绑定text-area -->
<div>
  <textarea v-model="article" cols="30" rows="10"></textarea>
  <h2>article当前的值是: {{article}}</h2>
</div>
```

v-model绑定checkbox

- 我们来看一下v-model绑定checkbox：单个勾选框和多个勾选框
- 单个勾选框：
  - v-model即为布尔值。
  - 此时input的value属性并不影响v-model的值。
- 多个复选框：
  - 当是多个复选框时，因为可以选中多个，所以对应的data中属性是一个数组。
  - 当选中某一个时，就会将input的value添加到数组中。

```
<!-- 2.1.单选框 -->
<div>
  <label for="agreement">
    <input id="agreement" type="checkbox" v-model="isAgree">同意协议
  </label>
  <h2>isAgree当前的值是: {{isAgree}}</h2>
</div>
```

```
<!-- 2.2.多选框 -->
<div>
  <label for="basketball">
    <input id="basketball" type="checkbox" value="basketball" v-model="hobbies">篮球
  </label>
  <label for="football">
    <input id="football" type="checkbox" value="football" v-model="hobbies">足球
  </label>
  <label for="tennis">
    <input id="tennis" type="checkbox" value="tennis" v-model="hobbies">网球
  </label>
  <h2>hobbies当前的值是: {{hobbies}}</h2>
</div>
```

v-model绑定radio

- v-model绑定**radio**，用于选择其中一项；

```
<!-- 3.绑定radio -->
<div>
  <label for="male">
    <input type="radio" id="male" v-model="gender" value="male">男
  </label>
  <label for="female">
    <input type="radio" id="female" v-model="gender" value="female">女
  </label>
  <h2>gender当前的值是: {{gender}}</h2>
</div>
```

v-model绑定select

- **和checkbox一样，select也分单选和多选两种情况。**
- **单选：只能选中一个值**
  - v-model绑定的是一个值；
  - 当我们选中option中的一个时，会将它对应的value赋值到fruit中；
- **多选：可以选中多个值**
  - v-model绑定的是一个数组；
  - 当选中多个值时，就会将选中的option对应的value添加到数组fruit中；

```
<div>
  <select v-model="fruit">
    <option value="apple">苹果</option>
    <option value="orange">橘子</option>
    <option value="banana">香蕉</option>
  </select>
  <h2>fruit当前的值是: {{fruit}}</h2>
</div>
```

```
<div>
  <select v-model="fruit" multiple size="3">
    <option value="apple">苹果</option>
    <option value="orange">橘子</option>
    <option value="banana">香蕉</option>
  </select>
  <h2>fruit当前的值是: {{fruit}}</h2>
</div>
```

v-model的值绑定

- **目前我们在前面的案例中大部分的值都是在template中固定好的：**
  - 比如gender的两个输入框值male、female；
  - 比如hobbies的三个输入框值basketball、football、tennis；
- 在真实开发中，我们的数据可能是来自服务器的，那么我们就可以先将值请求下来，绑定到data返回的对象中，再通过v-bind来进行值的绑定，这个过程就是**值绑定**。
  - 这里不再给出具体的做法，因为还是v-bind的使用过程。

v-model修饰符 - lazy

- **lazy修饰符是什么作用呢?**
  - 默认情况下，v-model在进行双向绑定时，绑定的是input事件，那么会在每次内容输入后就将最新的值和绑定的属性进行同步；
  - 如果我们在v-model后跟上lazy修饰符，那么会将绑定的事件切换为 change 事件，只有在提交时（比如回车）才会触发；

```
<template id="my-app">
  <input type="text" v-model.lazy="message">
  <h2>{{message}}</h2>
</template>
```

v-model修饰符 - number

- 我们先来看一下v-model绑定后的值是什么类型的：
  - message总是string类型，即使在我们设置type为number也是string类型；

```
<template id="my-app">
  <!-- 类型 -->
  <input type="text" v-model="message">
  <input type="number" v-model="message">
  <h2>{{message}}</h2>
</template>
```

- 如果我们希望转换为**数字类型**，那么可以使用 **.number 修饰符**：

```
<input type="text" v-model.number="score">
```

- 另外，在我们进行**逻辑判断**时，如果是一个**string类型**，在可以转化的情况下**会进行隐式转换**的：
  - 下面的score在进行判断的过程中会进行隐式转化的；

```
const score = "100";
if (score > 90) {
  console.log("优秀");
}
console.log(typeof score);
```

v-model修饰符 - trim

- 如果要自动过滤用户输入的守卫空白字符，可以给v-model添加 **trim 修饰符**：

```
<template id="my-app">
  <!-- 去除空格 -->
  <input type="text" v-model.trim="message">
</template>
```

v-mode组件上使用

- **v-model也可以使用在组件上，Vue2版本和Vue3版本有一些区别。**
  - 具体的使用方法，后面讲组件化开发再具体学习。