第十一课 组件基础

学习目录

- **组件含义**
- **组件基础用法**
- **组件通信**
- **组件插槽**
- **学习计划表每条计划组件**

一．组件含义

在之前课程中我们编写的学习计划表中，对于每条学习计划，我们是用一套 html 标签代码来表示的，其实在 vue 中，我们可以把这种通用的 html 标签代码封装成一个基于 vue 的组件，在 vue 实例对象的作用范围内任意使用即可。

组件在 vue 中其实相当于一个树形结构一样，在这个树形结构中，组件可以相互嵌套使用，不同的组件有着不同的作用，比如我们可能做一个网站，需要登录框、顶部导航栏、新闻列表等区域，这些区域都可以用组件的形式来封装，这样你在其他地方也就可以重复使用，在每次使用的时候传入不同的组件数据就可以得到不同的组件表现形式和数据显示形式，这实际上就是基于 vue 的组件化开发方法。

二．组件基础用法

要封装 vue 的一个组件，前提是必须在 vue 中注册这个组件，使用 Vue.component 这个方法可以在全局下注册并编写这个 vue 组件。

注册组件

```
Vue.component('custom-component', {
  data: function () {
    return {
      num:100
    }
  },
  template: '<div>{{num}}</div>'
})
```

使用组件

```
<div id="app">
  <custom-component></custom-component>
</div>
```

因为组件其实也是 vue 实例对象，只不过组件是可以复用的 vue 实例对象，所以它们与 new Vue 使用相同的选项，比如 data、computed、watch、methods 以及生命周期钩子函数等。但是他们不需要根元素作用范围的 el 选项，因为只要是组件使用的地方肯定是有根元素做好绑定了。

另外组件中的 data 是以函数返回值的形式出现的，这是因为组件在使用的时候都是独立的 vue 实例对象，因此它们要维护的 data 数据都是属于自身组件的 data 数据，所以使用函数返回值的形式可以保证组件实例对象的 data 数据是一个被返回对象的独立的拷贝。

另外在一个组件中，只能有一个单一的根元素，也就是说组件中的所有元素都是被包含在一个根元素下面的。

这样会报错

```
<h3>{{ title }}</h3>
<div>{{content}}</div>
```

这样用一个根元素包含其他所有的元素就不会报错了

```
<div>
    <h3>{{ title }}</h3>
    <div>{{content}}</div>
</div>
```

其实组件的注册可以由全局注册，也可以由局部注册，全局注册的组件可以在全局任何根实例对象的作用范围下使用，局部注册当然只能在注册的组件中使用，在目前基础课程中我们先掌握全局注册就可以了，在之后的进阶课程中我们在说局部注册组件问题。

三．组件通信

在我们做好了组件并在某个 vue 的根实例对象中使用了这个组件之后，下面我们就要考虑另一个问题了，我们如何把当前根实例对象的数据传递给子组件来使用，子组件又如何把它内部发生的变化告诉根实例对象呢？

这里我们先把根实例对象称为父组件，把在它里面的组件称为子组件。

父组件向子组件传递数据

prop 单向数据传递，prop 的意思是我们可以在子组件上注册的一些自定义 attribute 属性。当一个数据需要通过一个 prop 属性传递给子组件的时候，这个父组件的 attribute 属性就会变成了那个子组件的一个 property 对象属性。

```
Vue.component('custom-title', {
  props: ['title'],
  template: '<div>{{ title }}</div>'
})
<div id="app">
    <custom-title :title="星星课堂"></custom-title>
</div>
```

attribute 其实是元素标签属性，property 是元素对象属性，就好比一个输入框的 value，我们可以用 getAttribute 或者 setAttribute 来获取和设置这个 value，输入框 input 的 value 的对象属性可以用 input.value 获取和更新，它的初始值与 attribute 中的赋值是保持一致的数据。

子组件自定义事件与父组件通信

子组件可以接收父组件的数据，同样的当子组件内部发生一些数据变化之后也可以传递给父组件，vue 给我们提供了一种自定义事件的方式，可以方便的把子组件的改变传递给父组件。

子组件可以在发生变化的使用使用 vue 实例对象的$emit()方法来传递数据给父组件。

```
<div id="app">
    <div>{{sum}}</div>
    <custom-num :first="num1" :secend="num2" @add="showSumFn"></custom-num>
</div>
<script>

    Vue.component('custom-num',{
        props:['first','secend'],
        data:function(){
            return {
                third:1000
            }
        },
        template:`<div>
            <div>{{first}}</div>
            <div>{{secend}}</div>
            <button @click="fn1">按钮</button>
        </div>`,
        methods:{
            fn1(){
                let newSum = this.first + this.secend + this.third;
```

```
                this.$emit('add',newSum);
            }
        }
    })

    new Vue({
        el:'#app',
        data:{
            num1:100,
            num2:500,
            sum:0
        },
        methods:{
            showSumFn(value){
                console.log(value);
                this.sum = value;
            }
        }
    });
</script>
```

另外在 html 模板中编写的组件，$emit 传入的自定义事件名称全部要小写，在.vue 文件的组件中不受限制，我们会在之后的课程中为大家介绍.vue 文件的 vue 组件。

四．组件插槽

当我们写好了子组件之后，我们可能还有一种需求，可能子组件在不同的父组件中需要有不同的数据内容显示，可能还要有不同的文档结构和样式布局，因此 vue 提供了 slot 插槽来满足这方面的需求。

```
Vue.component('custom-num', {
  template: `
    <div>
      <h3>数字内容</h3>
      <slot></slot>
    </div>
})
<custom-num>
    <div class="one">100</div>
</custom-num>
<custom-num>
    <div class="two">300</div>
</custom-num>
<custom-num>
    <div class="three">600</div>
</custom-num>
```

如果插槽需要用的地方太多的话，这里还可以指定插槽的名称，在父组件中我们就可以向子组件的指定位置来插入我们想要放入子组件的内容了，这个我们在之后的课程在说。

五．学习计划表每条计划组件

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
    <style>
        *{
            margin:0px;
            padding:0px;
        }
        button{
            padding:10px 15px;
        }
        #app{
            width:100%;
            text-align: center;
        }
        .wrap{
```

```
        display: inline-block;
        width:600px;
        background: orange;
        text-align: left;
    }
    .inputPlanWrap{
        padding:30px 0px;
        border:1px solid #ccc;
        text-align: center;
    }
    .inputPlanWrap input{
        width:200px;
        height:40px;
        text-indent: 10px;
    }
    .studyPlan{
        text-align: right;
        padding:15px;
    }
    .planItem{
        display: flex;
        justify-content: space-around;
```

```
            border-bottom:1px solid #ccc;
            padding:10px 0px;
            align-items: center;
        }
        .deleteBtn{
            width:30px;
            height:30px;
        }
        .submitBtnWrap{
            padding:10px 0px;
            border-bottom:1px solid #ccc;
        }
        .totalWrap{
            padding:10px 0px;
        }
        .completedActive{
            background: #fff;
            color:orange;
        }
    </style>
    <script src="./vue.js"></script>
</head>
```

```
<body>
    <div id="app">
        <div class="wrap">
            <div class="inputPlanWrap">
                <input type="text" placeholder="请输入您的学习计划" v-model="planValue" />
                <button @click="submitPlanFn">提交</button>
            </div>
            <div class="studyPlan">
                <plan-item v-for="(item,index) in planListArr" :key="item.id" :id="item.id" :plan="item.plan" :status="item.status" @delete="deleteFn"></plan-item>
                <div class="submitBtnWrap">
                    <button @click="submitCompleteFn">确定完成</button>
                </div>
                <div class="totalWrap">
                    <span>学习计划的总数量: {{planListArr.length}}</span>
                    <span>学习计划的完成数量: {{completedArr.length}}</span>
                    <span>学习计划的剩余数量: {{planListArr.length - completedArr.length}}</span>
```

```
            </div>
        </div>
    </div>
</div>
<script>
    Vue.component('planItem',{
        props: ['id','plan','status'],
        data:function(){
            return {
                checkedArr:[],
            }
        },
        computed:{
            formatStatus(){
                return function(status){
                    return status == false ? '未完成' : '已完成';
                }
            }
        },
        template:`<div class="planItem">
                <span>id:{{id}}</span>
                <span>学习计划：{{plan}}</span>
```

```
                    <span>完成状态：{{formatStatus(status)}}</span>
                    <input type="checkbox" name="" id="" :value="id"
v-model="checkedArr" v-if="!status" />
                    <img src="./deleteIcon.png" class="deleteBtn" alt=""
v-if="status" @click="deleteItemFn(id)" />
                </div>`,
            methods:{
                deleteItemFn(id){
                    this.$emit('delete',id);
                }
            }
        })
        new Vue({
            el:'#app',
            data:{
                planListArr:[],
                planValue:'',
                completedArr:[]
            },
            created(){
                this.planListArr = this.getDataFn('planListArr');
                this.updateCountFn();
```

```
},
methods:{
    updateCountFn(){
        this.planListArr.map((item,index) => {
            if(item.status){
                this.completedArr.push(item.id);
            }
        });
    },
    setDataFn(key,value){
        localStorage.setItem(key,JSON.stringify(value));
    },
    getDataFn(key){
        return JSON.parse(localStorage.getItem(key));
    },
    submitPlanFn(){
        if(this.planValue == ''){
            alert('学习计划不能为空');
            return;
        }
        let obj = {
            id:Math.random() * 100 + new Date().getDate(),
```

```
                plan:this.planValue,
                status:false
            };
            this.planListArr.push(obj);
            this.planValue = '';
            this.setDataFn('planListArr',this.planListArr);
        },
        submitCompleteFn(){
            if(this.checkedArr.length == 0){
                alert('你还没有选择要完成的学习计划');
                return;
            }
            this.planListArr.map((item,index) => {
                this.checkedArr.map((checkedItem,checkedIndex)
=> {
                    if(item.id == checkedItem){
                        item.status = true;
                        this.completedArr.push(item.id);
                    }
                });
            });
            this.checkedArr = [];
```

```
                this.setDataFn('planListArr',this.planListArr);
            },
            deleteFn(id){
                let completedIndex = this.completedArr.findIndex((item) => item.id == id);
                this.completedArr.splice(completedIndex,1);
                this.planListArr = this.planListArr.filter((item,index) => {
                    return item.id != id;
                });
                this.checkedArr = [];
                this.setDataFn('planListArr',this.planListArr);
            }
        }
    });
</script>
</body>
</html>
```

谢谢观看！如果觉得课程还不错的话，记得给个好评！

我是星星课堂老师：周小周