第十六课 表单验证组件

学习目录

- ➢ **表单验证组件基础结构**
- ➢ **表单验证组件开发原理**
- ➢ **表单验证组件开发过程**

一．表单验证组件基础结构

```
 .inputWrap{
       margin:30px 0px;
 }
  <form id="formWrap">
       <div class="inputWrap">
            <label>用户名：</label>
            <input type="text" placeholder="请输入用户名" name="username" />
       </div>
       <div class="inputWrap">
            <label>密码：</label>
            <input type="password" placeholder="请输入密码" name="password"
/>
       </div>
       <div class="inputWrap">
```

```
            <label>手机号码：</label>
            <input type="text" placeholder="请输入手机号码" name="number" />
        </div>
        <div class="inputWrap">
            <label>昵称：</label>
            <input type="text" placeholder="请输入昵称" name="nickname" />
        </div>
        <div class="inputWrap">
            <label>邮箱：</label>
            <input type="text" placeholder="请输入邮箱" name="email" />
        </div>
        <button type="button" name="submit">提交数据</button>
    </form>
```

二．表单验证组件开发原理

这个表单验证组件主要是功能组件，当然肯定也会有 ui 结构与样式，同时我们这个表单验证组件是把验证反馈和验证规则划分成了两个构造函数来做，这样做的好处是把他们的职责划分的更清晰，保证在后期维护的时候只需要关注对应的功能修改即可，这也符合一定的设计原则中的单一职责原则，一个对象或者方法最多只有一个引起他变化的地方。

这个表单验证组件主要是为了加强我们对组件化开发实用性的学习，让我们能够利用学习到的组件化开发方法根据实际开发需要来编写代码，不用拘泥于形式，从而让组件化开发的效率最大化，同时我们的课程重点讲解的是组件化开发的知识，不会过多关注特别细致的表单验证规则，有兴趣的同学可以根据自己的需要进行相关扩展，方便自己使用。

三．表单验证组件开发过程

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>第十六课 表单验证组件</title>
    <style>
        .inputWrap{
            margin:30px 0px;
        }
    </style>
</head>

<body>
    <form class="formWrap">
        <div class="inputWrap">
            <label>用户名：</label>
            <input type="text" placeholder="请输入用户名" name="username" />
        </div>
        <div class="inputWrap">
```

```
                <label>密码：</label>

                <input type="password" placeholder="请输入密码" name="password"
/>

            </div>

            <div class="inputWrap">

                <label>手机号码：</label>

                <input type="text" placeholder="请输入手机号码" name="number" />

            </div>

            <div class="inputWrap">

                <label>昵称：</label>

                <input type="text" placeholder="请输入昵称" name="nickname" />

            </div>

            <div class="inputWrap">

                <label>邮箱：</label>

                <input type="text" placeholder="请输入邮箱" name="email" />

            </div>

            <button type="button" name="submit">提交数据</button>

        </form>

        <script>


            function FormValidation(){

                this.formWrap = null;
```

```
    this.rules = null;

    this.formLabel = null;

    this.submitForm = [];

    this.submitData = {};

}

FormValidation.prototype.init = function(){

    this.formWrap = document.getElementsByClassName('formWrap')[0];

}

FormValidation.prototype.bindInput = function(option){

    this.submitForm.push(option);

    this.eventCenter(option);

}

FormValidation.prototype.eventCenter = function(option){

    var that = this;

    this.formWrap[option.label].onblur = function(){

        that.blurFn(this,option);

    }

    this.formWrap['submit'].onclick = function(){

        that.submitFn();

    }

}

FormValidation.prototype.blurFn = function(_this,option){
        var label = option.label;

        var value = _this.value;

        var result = this.rules.formRuleFn(value,option.rules);

        this.formLabel = _this.parentElement;

        this.updateError(result);

        this.submitData[label] = value;

}

FormValidation.prototype.submitFn = function(){

        for(var i=0;i<this.submitForm.length;i++){

                var formElement = this.formWrap[this.submitForm[i].label];

                this.blurFn(formElement,this.submitForm[i]);

        }

        console.log(this.submitData);

}

FormValidation.prototype.creatError = function(result){

        var resultWrap = document.createElement('span');

        resultWrap.setAttribute('class','resultWrap');

        resultWrap.innerHTML = result.join(',');

        this.formLabel.appendChild(resultWrap);

}

FormValidation.prototype.removeError = function(){

        var resultWrapClass = this.getFormLabel().getAttribute('class');
        if(resultWrapClass == 'resultWrap'){
            return true;
        }else{
            return false;
        }
    }
    FormValidation.prototype.getFormLabel = function(){
        var formLabelWrap = this.formLabel.children;
        var resultWrapLength = formLabelWrap.length;
        return formLabelWrap[resultWrapLength - 1];
    }
    FormValidation.prototype.replaceResultWrap = function(result){
        var resultWrap = this.getFormLabel();
        var text = result.join(',');
        resultWrap.innerHTML = text;
    }
    FormValidation.prototype.updateError = function(result){
        if(!this.removeError()){
            this.creatError(result);
        }else{
            this.replaceResultWrap(result);
        }
    }




    function Rules(){
        var that = this;
        this.selectFn = {
            'required':this.required.bind(that),
            'minLength':this.minLength.bind(that),
            'maxLength':this.maxLength.bind(that),
            'checkPassword':this.checkPassword.bind(that),
            'checkNumber':this.checkNumber.bind(that),
            'checkEmail':this.checkEmail.bind(that),
        }
    }
    Rules.prototype.formRuleFn = function(value,rules){
        var result = [];
        for(var i=0;i<rules.length;i++){
            for(var rule in rules[i]){
                if(rule != 'errorInfo' && this.selectFn[rule](value,rules[i]) !==
true){
                    result.push(this.selectFn[rule](value,rules[i]));
                    }
                }
            }
            return result;
        }
        Rules.prototype.required = function(value,rule){
            return /^\s*$/g.test(value) ? rule.errorInfo : true;
        }
        Rules.prototype.minLength = function(value,rule){
            return value.length < rule.minLength ? rule.errorInfo : true;
        }
        Rules.prototype.maxLength = function(value,rule){
            return value.length > rule.maxLength ? rule.errorInfo : true;
        }
        Rules.prototype.checkPassword = function(value,rule){
            var result = true;
            if(rule.checkPassword == 1){
                result = !(/^[\w+]/).test(value) ? rule.errorInfo : true;
            }else if(rule.checkPassword == 2){
                result = !(/(?=.*[0-9])(?=.*[a-zA-Z]).//).test(value) ? rule.errorInfo :
true;
            }else{
```

```
                        result = !(/(?=.*[0-9])(?=.*[a-zA-Z])(?=.*[^0-9a-zA-Z])/).test(value) ?
rule.errorInfo : true;
                }
                return result;
            }
            Rules.prototype.checkNumber = function(value,rule){
                return !(/^1[3456789]\d{9}$/.test(value)) ? rule.errorInfo : true;
            }
            Rules.prototype.checkEmail = function(value,rule){
                return      !(/^\w+\@+[a-zA-Z0-9]+\.(com|cn|net)$/.test(value))      ?
rule.errorInfo : true;
            }

            var fv1 = new FormValidation();
            fv1.init();

            fv1.bindInput({
                label:'username',
                rules:[
                    {required:true,errorInfo:'用户名不能为空'},
                    {minLength:3,errorInfo:'最小长度不能小于 3 位'},
                    {maxLength:9,errorInfo:'最长长度不能大于 9 位'},
        ]
});
fv1.bindInput({
        label:'password',
        rules:[
                {required:true,errorInfo:'密码不能为空'},
                {minLength:6,errorInfo:'最少 6 位字符'},
                {maxLength:15,errorInfo:'最多 15 位字符'},
                {checkPassword:3,errorInfo:'请输入正确的密码格式'}
        ]
});
fv1.bindInput({
        label:'number',
        rules:[
                {required:true,errorInfo:'手机号码不能为空'},
                {minLength:5,errorInfo:'最小长度不能小于 5 位'},
                {checkNumber:true,errorInfo:'请输入正确的手机号码'},
        ]
});
fv1.bindInput({
        label:'nickname',
        rules:[
                {required:true,errorInfo:'昵称不能为空'}
            ]
        });
        fv1.bindInput({
            label:'email',
            rules:[
                {required:true,errorInfo:'邮箱不能为空'},
                {checkEmail:true,errorInfo:'请输入正确的邮箱格式'},
            ]
        });

        fv1.rules = new Rules();

    </script>
</body>

</html>
```

谢谢观看！

我是星星课堂老师：周小周