jQuery应用一之验证插件validate的使用

JavaScript 崔庆才 3年前 (2015-04-04) 6487浏览 0评论

综述

validate是一个用来验证表单提交的插件，应用十分广泛，具有如下的几个功能

- 自带了基本的验证规则
- 提供了丰富的验证信息提示功能
- 多种事件触发验证
- 自定义验证规则

下面我们就来感受一下这个插件的强大之处吧

插件下载

在这里我们需要用到的插件文件有

- jquery.validate.js
- jquery.validate.messages_cn.js
- jquery.min.js

一个是表单验证的主文件，另一个是设置中文提示的文件。

实例引入

我们先用一个小例子来感受一下使用 validate 插件的便捷之处，这个例子中加入了表单合法性验证和错误提示，代码如下

```
<!DOCTYPE>
<html>
<head>
    <title>validate验证插件</title>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.min.js">
    </script>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.validate.js">
    </script>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.validate.messages_cn.js">
    </script>
    <link type="text/css" rel="stylesheet" href="http://res.cuiqingcai.com/jqplugins/validate/style.css"></link>
    <script type="text/javascript">
        $(function() {
            $("#frmV").validate(
              {
                  /*自定义验证规则*/
                  rules: {
                      username: { required: true, minlength: 6 },
                      email: { required: true, email: true }
                  },
                  /*错误提示位置*/
                  errorPlacement: function(error, element) {
                      error.appendTo(element.siblings("span"));
                  }
              }
            );
        });
    </script>
</head>
<body>
     <form id="frmV" method="get" action="#">
     <div class="divFrame">
         <div class="divTitle">
              请输入下列资料
         </div>
         <div class="divContent">
              <div>
              用户名：<br />
              <input id="username" name="username"
                     type="text" class="txt" />
              <font color="red">*</font><br />
              <span></span>
              </div>
              <div>
              邮箱：<br />
              <input id="email" name="email"
                     type="text" class="txt" />
              <font color="red">*</font><br />
```

```
                <span></span>
                </div>
            </div>
            <div class="divBtn">
                <input id="sbtUser" type="submit"
                    value="提交" class="btn" />
            </div>
        </div>
        </form>
    </body>
    </html>
```

运行结果如下

在这里我们定义了 rules 来控制表单的合法性，通过 errorPlacement 来控制错误的输出位置。

校验规则

下面我们详细说一下关于rules的相关知识，将校检规则总结如下

序号	规则	描述
1	required:true	必须输入的字段。
2	remote:”check.php”	使用 ajax 方法调用 check.php 验证输入值。
3	email:true	必须输入正确格式的电子邮件。
4	url:true	必须输入正确格式的网址。
5	date:true	必须输入正确格式的日期。日期校验 ie6 出错，慎用。
6	dateISO:true	必须输入正确格式的日期（ISO），例如：2009-06-23，1998/01/22。只验证格式，不验证有效性。
7	number:true	必须输入合法的数字（负数，小数）。
8	digits:true	必须输入整数。
9	creditcard:	必须输入合法的信用卡号。
10	equalTo:”#field”	输入值必须和 #field 相同。
11	accept:	输入拥有合法后缀名的字符串（上传文件的后缀）。
12	maxlength:5	输入长度最多是 5 的字符串（汉字算一个字符）。
13	minlength:10	输入长度最小是 10 的字符串（汉字算一个字符）。
14	rangelength:[5,10]	输入长度必须介于 5 和 10 之间的字符串（汉字算一个字符）。
15	range:[5,10]	输入值必须介于 5 和 10 之间。
16	max:5	输入值不能大于 5。
17	min:10	输入值不能小于 10。

比如我们针对 email 这个表单就可以定义为

```
email: { required: true, email: true }
```

针对url的这个输入表单就可以定义为

```
url: { required: true, url: true }
```

以上便是校验规则的相关内容。

消息提示

在 jquery.validate.js 这个文件中，定义了默认的消息提示，不过它是英文的提示，默认的提示如下

```
messages: {
    required: "This field is required.",
    remote: "Please fix this field.",
    email: "Please enter a valid email address.",
    url: "Please enter a valid URL.",
    date: "Please enter a valid date.",
    dateISO: "Please enter a valid date (ISO).",
    dateDE: "Bitte geben Sie ein gültiges Datum ein.",
    number: "Please enter a valid number.",
    numberDE: "Bitte geben Sie eine Nummer ein.",
    digits: "Please enter only digits",
    creditcard: "Please enter a valid credit card number.",
    equalTo: "Please enter the same value again.",
    accept: "Please enter a value with a valid extension.",
    maxlength: $.validator.format("Please enter no more than {0} characters."),
    minlength: $.validator.format("Please enter at least {0} characters."),
    rangelength: $.validator.format("Please enter a value between {0} and {1} characters long."),
    range: $.validator.format("Please enter a value between {0} and {1}."),
    max: $.validator.format("Please enter a value less than or equal to {0}."),
    min: $.validator.format("Please enter a value greater than or equal to {0}.")
},
```

比如，如果遇到 email 校验有问题，那么便会提示

```
Please enter a valid email address
```

不过我们通过引入 jquery.validate.messages_cn.js 这个文件，写入了如下代码，将默认的提示修改为中文

```
jQuery.extend(jQuery.validator.messages, {
        required: "必选字段",
        remote: "请修正该字段",
        email: "请输入正确格式的电子邮件",
        url: "请输入合法的网址",
        date: "请输入合法的日期",
        dateISO: "请输入合法的日期 (ISO).",
        number: "请输入合法的数字",
        digits: "只能输入整数",
        creditcard: "请输入合法的信用卡号",
        equalTo: "请再次输入相同的值",
        accept: "请输入拥有合法后缀名的字符串",
        maxlength: jQuery.format("请输入一个长度最多是 {0} 的字符串"),
        minlength: jQuery.format("请输入一个长度最少是 {0} 的字符串"),
        rangelength: jQuery.format("请输入一个长度介于 {0} 和 {1} 之间的字符串"),
        range: jQuery.format("请输入一个介于 {0} 和 {1} 之间的值"),
        max: jQuery.format("请输入一个最大为 {0} 的值"),
        min: jQuery.format("请输入一个最小为 {0} 的值")
});
```

当然，以上的设置都是默认的提示，我们还可以通过 messages 来设置提示，举一个小例子，加入 messages 选项

```
<!DOCTYPE>
<html>
<head>
    <title>validate验证插件</title>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.min.js">
    </script>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.validate.js">
    </script>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.validate.messages_cn.js">
    </script>
    <link type="text/css" rel="stylesheet" href="http://res.cuiqingcai.com/jqplugins/validate/style.css">
    <script type="text/javascript">
        $(function() {
            $("#frmV").validate(
              {
                  /*自定义验证规则*/
                  rules: {
                      username: { required: true, minlength: 6 },
                      email: { required: true, email: true }
                  },
                  /*错误提示位置*/
                  errorPlacement: function(error, element) {
                      error.appendTo(element.siblings("span"));
                  },
                  messages: {
                      username: { required: "请输入姓名", minlength: "长度不可小于6" },
                      email: { required: "请输入电子邮件", email: "请输入正确格式" }
                  }
              }
            );
        })
    </script>
</head>
<body>
     <form id="frmV" method="get" action="#">
     <div class="divFrame">
         <div class="divTitle">
              请输入下列资料
         </div>
         <div class="divContent">
              <div>
              用户名：<br />
              <input id="username" name="username"
                     type="text" class="txt" />
              <font color="red">*</font><br />
              <span></span>
              </div>
              <div>
```

```
            邮箱：<br />
            <input id="email" name="email"
                   type="text" class="txt" />
            <font color="red">*</font><br />
            <span></span>
            </div>
        </div>
        <div class="divBtn">
            <input id="sbtUser" type="submit"
                   value="提交" class="btn" />
        </div>
    </div>
    </form>
</body>
</html>
```

运行结果如下

失败验证

```
 errorPlacement: function(error, element) {
error.appendTo(element.siblings("span"));
},
```

我们用 errorPlacement 来处理验证失败后的处理，方法有两个参数，一个是error，一个是element。

其中error是字符串，保存了messages中返回的错误信息，element是验证失败的input元素。

比如上面这一句

```
error.appendTo(element.siblings("span"));
```

就代表把错误加入到input元素同级的span元素中，从而在标签内部显示错误的内容。

其他的情况我们可以灵活处理。

成功验证

有失败就有成功，在这里我们可以用一个函数来实现成功的验证

```
success: function(label) {
label.html("OK");
}
```

这里的label指的是发生错误时那个标签，就是上面例子中的span，通过html()方法可以实现标签内容的变化。例如下面的例子

```
<!DOCTYPE>
<html>
<head>
    <title>validate验证插件</title>
    <meta charset="utf-8"/>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.min.js">
    </script>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.validate.js">
    </script>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.validate.messages_cn.js">
    </script>
    <link type="text/css" rel="stylesheet" href="http://res.cuiqingcai.com/jqplugins/validate/style.css">
    <script type="text/javascript">
        $(function() {
            $("#frmV").validate(
              {
                  /*自定义验证规则*/
                  rules: {
                      username: { required: true, minlength: 6 },
                      email: { required: true, email: true }
                  },
                  /*错误提示位置*/
                  errorPlacement: function(error, element) {
                      error.appendTo(element.siblings("span"));
                  },
                  messages: {
                      username: { required: "请输入姓名", minlength: "长度不可小于6" },
                      email: { required: "请输入电子邮件", email: "请输入正确格式" }
                  },
                  success: function(label) {
                      label.html("OK");
                  }
              }
            );
        })
    </script>
</head>
<body>
     <form id="frmV" method="get" action="#">
     <div class="divFrame">
         <div class="divTitle">
              请输入下列资料
         </div>
         <div class="divContent">
              <div>
              用户名：<br />
              <input id="username" name="username"
                     type="text" class="txt" />
```

```
                <font color="red">*</font><br />
                <span>呵呵</span>
                </div>
                <div>
                邮箱：<br />
                <input id="email" name="email"
                       type="text" class="txt" />
                <font color="red">*</font><br />
                <span></span>
                </div>
          </div>
          <div class="divBtn">
                <input id="sbtUser" type="submit"
                       value="提交" class="btn" />
          </div>
      </div>
      </form>
</body>
</html>
```

上面就是验证成功之后的效果，在相应提示的地方会显示OK。

异步验证

有时候我们需要用到异步验证，我们可以在rules中加入remote进行远程验证，例子如下

```
<!DOCTYPE>
<html>
<head>
    <title>validate验证插件</title>
    <meta charset="utf-8"/>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.min.js">
    </script>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.validate.js">
    </script>
    <script type="text/javascript"
            src="http://res.cuiqingcai.com/js/jquery.validate.messages_cn.js">
    </script>
    <link type="text/css" rel="stylesheet" href="http://res.cuiqingcai.com/jqplugins/validate/style.css">
    <script type="text/javascript">
        $(function() {
            $("#frmV").validate(
                {
                    /*自定义验证规则*/
                    rules: {
                        username: { required: true, minlength: 6 },
                        phone: {
                            required: true,
                            remote:{
                                url: "check_phone.php",      //后台处理程序
                                type: "post",                //数据发送方式
                                dataType: "json",            //接受数据格式
                                data: {                      //要传递的数据
                                    phone: function() {
                                        return $("#phone").val();
                                    }
                                }
                            }
                        }
                    },
                    /*错误提示位置*/
                    errorPlacement: function(error, element) {
                      error.appendTo(element.siblings("span"));
                    },
                    messages: {
                      username: { required: "请输入姓名", minlength: "长度不可小于6" },
                      phone: { required: "请输入电话", remote: "请输入正确格式" }
                    },
                    success: function(label) {
                      label.html("OK");
                    }
                }
            );
        })
    </script>
```

```
</head>
<body>
     <form id="frmV" method="get" action="#">
     <div class="divFrame">
          <div class="divTitle">
               请输入下列资料
          </div>
          <div class="divContent">
               <div>
               用户名：<br />
               <input id="username" name="username"
                      type="text" class="txt" />
               <font color="red">*</font><br />
               <span></span>
               </div>
               <div>
               电话号码：<br />
               <input id="phone" name="phone"
                      type="text" class="txt" />
               <font color="red">*</font><br />
               <span></span>
               </div>
          </div>
          <div class="divBtn">
               <input id="sbtUser" type="submit"
                      value="提交" class="btn" />
          </div>
     </div>
     </form>
</body>
</html>
```

PHP处理程序，注意这里的返回值只能是true或者false，并且需要加引号。

```
<?php
    $phone = $_POST['phone'];
    if((strlen($phone) != 11) || !(preg_match("/13[0123456789]{1}\d{8}|15[012356789]\d{8}|18[0123456789]\d{8}|17[0678]\d{8}|14[57]\d{8}/",$phone))){
        echo "false";
     }else{
        echo "true";
     }
?>
```

演示如下

上面就是进行ajax异步验证的处理方式

自定义方法

有时候我们需要自定义一些验证方法，我们就需要用到addMethod方法，介绍如下

addMethod(name,method,message)方法

> 参数name 是添加的方法的名字
>
> 参数method是一个函数,接收三个参数(value,element,param) value 是元素的值,element是元素本身
>
> param是参数,我们可以用addMethod 来添加除built-in Validation methods 之外的验证方法

例如手机号码的验证如下

```
$.validator.addMethod("phone",function(value,element,params){
        if((value.length != 11) || (!value.match(/^(13[0-9]|14[5|7]|15[0|1|2|3|5|6|7|8|9]|17[0|6|7|8]|18[0-9])\d{8}$/))){
            return false;
         }else{
            return true;
        }
    },"请输入正确的手机号");
```

使用时如下

```
rules:{
            phone:{
                required:true,phone:true
            },
        },
```

有一个字段,只能输一个字母,范围是a-f,写法如下

```
$.validator.addMethod("af",function(value,element,params){
    if(value.length>1){
        return false;
    }
    if(value>=params[0] && value<=params[1]){
        return true;
    }else{
        return false;
    }
},"必须是一个字母,且a-f");
```

使用时如下

```
rules:{
            username:{ af:["a","f"] }
        },
```

以上便是自定义验证方法的方式

DeBug模式

开启DeBug模式后，不会进行提交，只需要在代码中加入

```
debug:true
```

即可，这样不论怎样，都不会提交表单，对于调试十分有用。

验证通过提交

在上面的例子中，我们没有设置表单验证通过之后才提交，通过加入以下代码，可以实现验证之后才提交的效果

```
submitHandler:function(form){
form.submit();
}
```

通过设置上面的内容，我们就可以避免验证不成功submit跳转了

忽略元素

有时候，我们想跳过某些元素不进行验证，可以通过加入如下代码来实现，举例如下

```
ignore:"input",
```

忽略所有input元素

```
ignore:"#username",
```

忽略id为username的元素

```
ignore:".input",
```

忽略所有class为input的元素

响应事件

在默认的响应事件是 submit 提交事件，我们还可以通过设置来改变事件的响应，比如失去焦点时验证等等，举例如下

Onubmit：类型 Boolean，默认 true，指定是否提交时验证。

```
$(".selector").validate({       onsubmit:false })
```

onfocusout：类型 Boolean，默认 true，指定是否在获取焦点时验证。

```
$(".selector").validate({     onfocusout:false })
```

onkeyup：类型 Boolean，默认 true，指定是否在敲击键盘时验证。

```
$(".selector").validate({    onkeyup:false })
```

onclick：类型 Boolean，默认 true，指定是否在鼠标点击时验证（一般验证 checkbox、radiobox）。

```
$(".selector").validate({    onclick:false })
```

focusInvalid：类型 Boolean，默认 true。提交表单后，未通过验证的表单（第一个或提交之前获得焦点的未通过验证的表单）会获得焦点。

```
$(".selector").validate({    focusInvalid:false })
```

focusCleanup：类型 Boolean，默认 false。当未通过验证的元素获得焦点时，移除错误提示（避免和 focusInvalid 一起使用）。

```
$(".selector").validate({    focusCleanup:true })
```

上面的响应事件一般不太常用，仅作了解即可。

总结

以上便是jQuery插件validate的用法，利用好这款插件对于编写将有极其大的帮助，希望大家能好好学习！

转载请注明：静觅 » jQuery应用一之验证插件validate的使用

想结交更多的朋友吗?

来进击的Coder瞧瞧吧

进击的Coder

QQ群号 99350970 立即加入

进击的Coder灌水太多?

这里是纯粹的技术领地

激进的Coder

QQ群号 627725766 立即加入

想找人聊天解闷？想要学习干货？

微信公众号进击的Coder为你打造

进击的Coder

微信公众号 扫一扫关注

您的支持是博主写作最大的动力，如果您喜欢我的文章，感觉我的文章对您有帮助，请狠狠点击下面的

我要小额赞助

jQuery JS

« 致那逝去的三月

jQuery应用二之邮箱下拉列表自动补全 »

JavaScript加密逻辑分析与Python模拟执行实现数据爬

[北京][14k-25k][PHP + 前端][两年经验]

BootStrap4提取并编译Flexbox Grid系统

Web安全学习一之XSS漏洞的利用

- JavaScript加密逻辑分析与Python模拟执行实现数据爬
- [北京][14k-25k][PHP + 前端][两年经验]
- BootStrap4提取并编译Flexbox Grid系统
- Web安全学习一之XSS漏洞的利用

- Python爬虫利器四之PhantomJS的用法
- JavaScript与jQuery基本用法总结
- jQuery易忽略的知识点总结
- 关于HTML内联元素一侧留白的浅谈