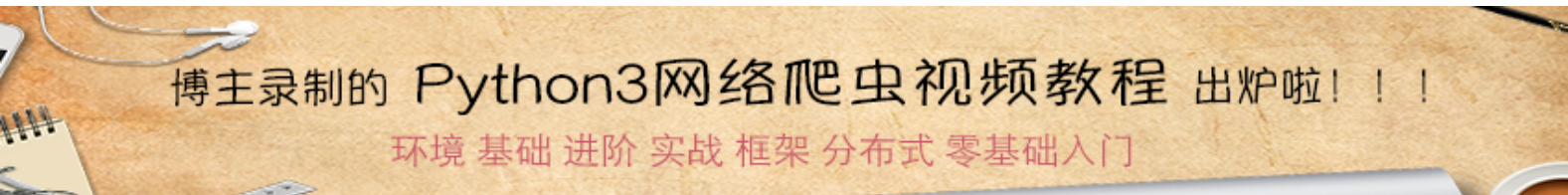


jQuery应用一之验证插件validate的使用

📖 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

```

1 <!DOCTYPE>
2 <html>
3 <head>
4     <title>validate验证插件</title>
5     <script type="text/javascript"
6         src="http://res.cuiqingcai.com/js/jquery.min.js">
7     </script>
8     <script type="text/javascript"
9         src="http://res.cuiqingcai.com/js/jquery.validate.js">
10    </script>
11    <script type="text/javascript"
12        src="http://res.cuiqingcai.com/js/jquery.validate.messages_cn.js">
13    </script>
14    <link type="text/css" rel="stylesheet" href="http://res.cuiqingcai.com/jqplugins/validate/styl
15e.css"></link>
16    <script type="text/javascript">
17        $(function() {
18            $("#frmV").validate(
19                {
20                    /*自定义验证规则*/
21                    rules: {
22                        username: { required: true, minlength: 6 },
23                        email: { required: true, email: true }
24                    },
25                    /*错误提示位置*/
26                    errorPlacement: function(error, element) {
27                        error.appendTo(element.siblings("span"));
28                    }
29                }
30            );
31        });
32    </script>
33 </head>
34 <body>
35     <form id="frmV" method="get" action="#">
36     <div class="divFrame">
37         <div class="divTitle">
38             请输入下列资料
39         </div>
40         <div class="divContent">
41             <div>
42                 用户名: <br />
43                 <input id="username" name="username"
44                     type="text" class="txt" />
45                 <font color="red">*</font><br />
46                 <span></span>
47             </div>
48             <div>
49                 邮箱: <br />
50                 <input id="email" name="email"
51                     type="text" class="txt" />
52                 <font color="red">*</font><br />

```

```
56         <span></span>
57     </div>
58 </div>
59 <div class="divBtn">
60     <input id="sbtUser" type="submit"
61         value="提交" class="btn" />
62 </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

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

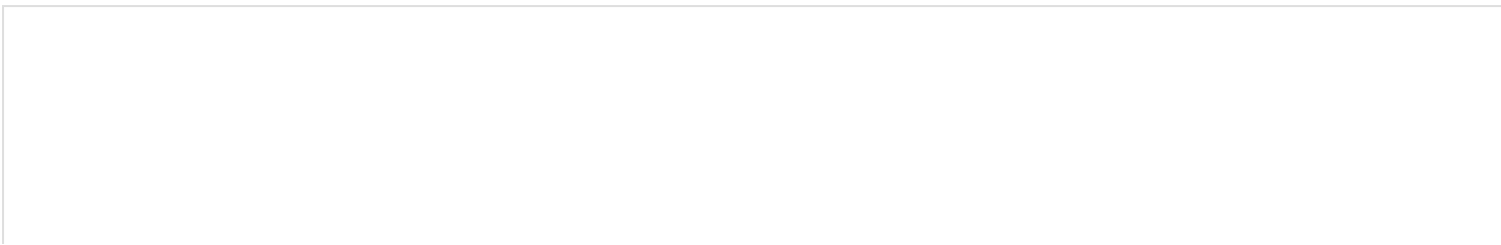
针对url的这个输入表单就可以定义为

```
1 | 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



```

1 messages: {
2     required: "This field is required.",
3     remote: "Please fix this field.",
4     email: "Please enter a valid email address.",
5     url: "Please enter a valid URL.",
6     date: "Please enter a valid date.",
7     dateISO: "Please enter a valid date (ISO).",
8     dateDE: "Bitte geben Sie ein gültiges Datum ein.",
9     number: "Please enter a valid number.",
10    numberDE: "Bitte geben Sie eine Nummer ein.",
11    digits: "Please enter only digits",
12    creditcard: "Please enter a valid credit card number.",
13    equalTo: "Please enter the same value again.",
14    accept: "Please enter a value with a valid extension.",
15    maxlength: $.validator.format("Please enter no more than {0} characters."),
16    minlength: $.validator.format("Please enter at least {0} characters."),
17    rangelength: $.validator.format("Please enter a value between {0} and {1} characters long."),
18    range: $.validator.format("Please enter a value between {0} and {1}."),
19    max: $.validator.format("Please enter a value less than or equal to {0}."),
20    min: $.validator.format("Please enter a value greater than or equal to {0}."),
21 },

```

比如，如果遇到 email 校验有问题，那么便会提示

```
1 | 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

```

1 jQuery.extend(jQuery.validator.messages, {
2     required: "必选字段",
3     remote: "请修正该字段",
4     email: "请输入正确格式的电子邮件",
5     url: "请输入合法的网址",
6     date: "请输入合法的日期",
7     dateISO: "请输入合法的日期 (ISO).",
8     number: "请输入合法的数字",
9     digits: "只能输入整数",
10    creditcard: "请输入合法的信用卡号",
11    equalTo: "请再次输入相同的值",
12    accept: "请输入拥有合法后缀名的字符串",
13    maxlength: jQuery.format("请输入一个长度最多是 {0} 的字符串"),
14    minlength: jQuery.format("请输入一个长度最少是 {0} 的字符串"),
15    rangelength: jQuery.format("请输入一个长度介于 {0} 和 {1} 之间的字符串"),
16    range: jQuery.format("请输入一个介于 {0} 和 {1} 之间的值"),
17    max: jQuery.format("请输入一个最大为 {0} 的值"),
18    min: jQuery.format("请输入一个最小为 {0} 的值")
19 });

```

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




```
1 <!DOCTYPE>
2 <html>
3 <head>
4     <title>validate验证插件</title>
5     <script type="text/javascript"
6         src="http://res.cuiqingcai.com/js/jquery.min.js">
7     </script>
8     <script type="text/javascript"
9         src="http://res.cuiqingcai.com/js/jquery.validate.js">
10    </script>
11    <script type="text/javascript"
12        src="http://res.cuiqingcai.com/js/jquery.validate.messages_cn.js">
13    </script>
14    <link type="text/css" rel="stylesheet" href="http://res.cuiqingcai.com/jqplugins/validate/style.css">
15    <script type="text/javascript">
16        $(function() {
17            $("#frmV").validate(
18                {
19                    /*自定义验证规则*/
20                    rules: {
21                        username: { required: true, minlength: 6 },
22                        email: { required: true, email: true }
23                    },
24                    /*错误提示位置*/
25                    errorPlacement: function(error, element) {
26                        error.appendTo(element.siblings("span"));
27                    },
28                    messages: {
29                        username: { required: "请输入姓名", minlength: "长度不可小于6" },
30                        email: { required: "请输入电子邮件", email: "请输入正确格式" }
31                    }
32                }
33            );
34        })
35    </script>
36 </head>
37 <body>
38     <form id="frmV" method="get" action="#">
39     <div class="divFrame">
40         <div class="divTitle">
41             请输入下列资料
42         </div>
43         <div class="divContent">
44             <div>
45                 用户名: <br />
46                 <input id="username" name="username"
47                     type="text" class="txt" />
48                 <font color="red">*</font><br />
49                 <span></span>
50             </div>
51             <div>
52                 <input type="password" name="password" />
53                 <font color="red">*</font><br />
54                 <span></span>
55             </div>
56             <div>
```

```

57         邮箱: <br />
58         <input id="email" name="email"
59             type="text" class="txt" />
60         <font color="red">*</font><br />
61         <span></span>
62     </div>
63 </div>
64 <div class="divBtn">
65     <input id="sbtUser" type="submit"
66         value="提交" class="btn" />
    </div>
</div>
</form>
</body>
</html>

```

运行结果如下

失败验证

```

1  errorPlacement: function(error, element) {
2  error.appendTo(element.siblings("span"));
3  },

```

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

```
1 | 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

```
1 | success: function(label) {  
2 |   label.html("OK");  
3 | }
```

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

```

1 <!DOCTYPE>
2 <html>
3 <head>
4     <title>validate验证插件</title>
5     <meta charset="utf-8"/>
6     <script type="text/javascript"
7         src="http://res.cuiqingcai.com/js/jquery.min.js">
8     </script>
9     <script type="text/javascript"
10         src="http://res.cuiqingcai.com/js/jquery.validate.js">
11     </script>
12     <script type="text/javascript"
13         src="http://res.cuiqingcai.com/js/jquery.validate.messages_cn.js">
14     </script>
15     <link type="text/css" rel="stylesheet" href="http://res.cuiqingcai.com/jqplugins/validate/style.css">
16     <script type="text/javascript">
17         $(function() {
18             $("#frmV").validate(
19                 {
20                     /*自定义验证规则*/
21                     rules: {
22                         username: { required: true, minlength: 6 },
23                         email: { required: true, email: true }
24                     },
25                     /*错误提示位置*/
26                     errorPlacement: function(error, element) {
27                         error.appendTo(element.siblings("span"));
28                     },
29                     messages: {
30                         username: { required: "请输入姓名", minlength: "长度不可小于6" },
31                         email: { required: "请输入电子邮件", email: "请输入正确格式" }
32                     },
33                     success: function(label) {
34                         label.html("OK");
35                     }
36                 }
37             );
38         })
39     </script>
40 </head>
41 <body>
42     <form id="frmV" method="get" action="#">
43     <div class="divFrame">
44     <div class="divTitle">
45         请输入下列资料
46     </div>
47     <div class="divContent">
48     <div>
49         用户名: <br />
50         <input id="username" name="username"
51             type="text" class="txt" />

```

```
57         <font color="red">*</font><br />
58         <span>呵呵</span>
59     </div>
60     <div>
61         邮箱: <br />
62         <input id="email" name="email"
63             type="text" class="txt" />
64         <font color="red">*</font><br />
65         <span></span>
66     </div>
67 </div>
68 <div class="divBtn">
69     <input id="sbtUser" type="submit"
70         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

```

1 <!DOCTYPE>
2 <html>
3 <head>
4   <title>validate验证插件</title>
5   <meta charset="utf-8"/>
6   <script type="text/javascript"
7     src="http://res.cuiqingcai.com/js/jquery.min.js">
8   </script>
9   <script type="text/javascript"
10     src="http://res.cuiqingcai.com/js/jquery.validate.js">
11   </script>
12   <script type="text/javascript"
13     src="http://res.cuiqingcai.com/js/jquery.validate.messages_cn.js">
14   </script>
15   <link type="text/css" rel="stylesheet" href="http://res.cuiqingcai.com/jqplugins/validate/styl
16 e.css">
17   <script type="text/javascript">
18     $(function() {
19       $("#frmV").validate(
20         {
21           /*自定义验证规则*/
22           rules: {
23             username: { required: true, minlength: 6 },
24             phone: {
25               required: true,
26               remote: {
27                 url: "check_phone.php", //后台处理程序
28                 type: "post", //数据发送方式
29                 dataType: "json", //接受数据格式
30                 data: { //要传递的数据
31                   phone: function() {
32                     return $("#phone").val();
33                   }
34                 }
35             }
36           },
37           /*错误提示位置*/
38           errorPlacement: function(error, element) {
39             error.appendTo(element.siblings("span"));
40           },
41           messages: {
42             username: { required: "请输入姓名", minlength: "长度不可小于6" },
43             phone: { required: "请输入电话", remote: "请输入正确格式" }
44           },
45           success: function(label) {
46             label.html("OK");
47           }
48         }
49       );
50     });
51   </script>
52
53
54
55
56

```

```

57 </head>
58 <body>
59     <form id="frmV" method="get" action="#">
60         <div class="divFrame">
61             <div class="divTitle">
62                 请输入下列资料
63             </div>
64             <div class="divContent">
65                 <div>
66                     用户名: <br />
67                     <input id="username" name="username"
68                         type="text" class="txt" />
69                     <font color="red">*</font><br />
70                     <span></span>
71                 </div>
72                 <div>
73                     电话号码: <br />
74                     <input id="phone" name="phone"
75                         type="text" class="txt" />
76                     <font color="red">*</font><br />
77                     <span></span>
78                 </div>
79             </div>
80             <div class="divBtn">
81                 <input id="sbtUser" type="submit"
82                     value="提交" class="btn" />
83             </div>
84         </div>
85     </form>
86 </body>
87 </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。

```

1 <?php
2     $phone = $_POST['phone'];
3     if((strlen($phone) != 11) || !(preg_match("/13[0123456789]{1}\d{8}|15[012356789]\d{8}|18[0123456
4 789]\d{8}|17[0678]\d{8}|14[57]\d{8}/", $phone))) {
5         echo "false";
6     }else{
7         echo "true";
8     }
9 ?>

```

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

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


使用时如下

```
1 rules:{
2     phone:{
3         required:true,phone:true
4     },
5 }
```

有一个字段,只能输一个字母,范围是a-f,写法如下

```
1 $.validator.addMethod("af",function(value,element,params){
2     if(value.length>1){
3         return false;
4     }
5     if(value>=params[0] && value<=params[1]){
6         return true;
7     }else{
8         return false;
9     }
10 },"必须是一个字母,且a-f");
```

使用时如下

```
1 rules:{
2     username:{ af:["a","f"] }
3 }
```

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

```
1 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

```
1 | submitHandler: function(form){  
2 |   form.submit();  
3 | }
```

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

```
1 | ignore: "input",
```

忽略所有input元素

```
1 | ignore: "#username",
```

忽略id为username的元素

```
1 | 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。

```
1 | $("<strong>.selector</strong>").validate({      onsubmit: false })
```

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

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

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

```
1 | $(".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) 。

```
1 | $(".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。

```
1 | $(".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 focus Invalid 一起使用) 。

```
1 | $(".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](#)

♡ 喜欢 (13)

or

🔗 分享 (0)

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

BootStrap4提取并编译Flexbox Grid系统

[北京][14k-25k][PHP + 前端][两年经验]

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](#)
