jQuery验证框架学习笔记

作者: koalaxyq http://koalaxyq.javaeye.com

a小猪的jQuery验证框架学习笔记

目 录

1. JQyery

1.1 jQuery验证框架（一）可选项 (jQuery validation)

发表时间: 2009-12-25

jQuery验证框架

```
<script type="text/javascript" src=js/jquery-1.3.2.min.js></script>
<script type="text/javascript" src=js/jquery.validate.pack.js></script>
<script type="text/javascript">
    $(document).ready(function(){
        $("#textForm").validate();
    });
</script>
<form class="cmxform" id="commentForm" method="get" action="">
    Name<input id="cname" name="name" size="25" class="required" minlength="2" />
        <input class="submit" type="submit" value="Submit"/>
</form>
```

此文谨以以上js片段开始介绍jQuery Validation。

验证从这个方法开始：**validate([options])**

一、可选项(options)

[1] debug 类型：Boolean 默认：false

说明：开启调试模式。如果为true，表单不会提交，而且会在控制台显示一些错误消息（需要Firebug或者Firebug lite）。当要阻止表单默认提交事件，尝试去开启它。

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

[2] submitHandler 类型：Callback 默认：default (native) form submit

说明：当表单通过验证，提交表单。回调函数有个默认参数form

```
$(".selector").validate({
   submitHandler: function(form) {
       // do other stuff for a valid form
```

```
        form.submit();
    }
})
```

[3] invalidHandler 类型：Callback

说明：当未通过验证的表单提交时，可以在该回调函数中处理一些事情。该回调函数有两个参数：第一个为一个事件对象，第二个为验证器（validator）

```
$(".selector").validate({
    invalidHandler: function(form, validator) {
      var errors = validator.numberOfInvalids();
      if (errors) {
        var message = errors == 1
          ? 'You missed 1 field. It has been highlighted'
          : 'You missed ' + errors + ' fields. They have been highlighted';
        $("div.error span").html(message);
        $("div.error").show();
      } else {
        $("div.error").hide();
      }
    }
 })
```

[4] ignore 类型：Seletor

说明：当进行表单验证时，过滤掉选择器所选择的表单。用了jQuery not方法（ not() ）。类型为submit和reset的表单总是被忽略的。

```
$("#myform").validate({
   ignore: ".ignore"
})
```

[5] rules 类型：Options 默认：rules are read from markup (classes, attributes, metadata)

说明：用户定义的键/值对规则。键为一个表单元素的name属性（或是一组单选/复选按钮）、值为一个简单的字符串或者由规则/参数对（rule/parameter）组成的一个对象。可以和 class/attribute/metadata 规则一起使用。每个规则可以指定一个依存的验证前提条件。

```
$(".selector").validate({
   rules: {
     // simple rule, converted to {required:true}
     name: "required",
     // compound rule
     email: {
       required: true,
       email: true
     }/*
     email: {
         depends: function(element) {
           return $("#contactform_email:checked")
         }
     }*/
   }
})
```

[6] messages 类型：Options 默认：验证方法默认使用的消息

说明：用户自定义的 键/值 对消息。键为一个表单元素的name属性，值为该表单元素将要显示的消息。该消息覆盖元素的title属性或者默认消息。消息可以是一个字符串或者一个回调函数。回调函数必须在验证器的作用域中调用，将规则参数作为回调函数的第一个参数，将该表单元素作为回调函数的第二个参数，且必须返回一个字符串类型的消息。

```
$(".selector").validate({
   rules: {
     name: "required",
     email: {
       required: true,
       email: true
     }
   },
   messages: {
```

```
    name: "Please specify your name",
    email: {
      required: "We need your email address to contact you",
      email: "Your email address must be in the format of name@domain.com"
    }
  }
})
```

[7] groups 类型：Options

说明：指定错误消息分组。一个组由一个任意的组名作为键，一个由空白符分割的表单元素name属性列表作为值。用errorPlacement定义组消息的存放位置。

```
$("#myform").validate({
  groups: {
    username: "fname lname"
  },
  errorPlacement: function(error, element) {
    if (element.attr("name") == "fname"
                || element.attr("name") == "lname" )
      error.insertAfter("#lastname");
    else
      error.insertAfter(element);
   },
   debug:true
 })
```

[8] onsubmit 类型：Boolean 默认：true

说明：提交时验证表单。当设置为false时，只能用其它的事件验证。

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

[9] onfocusout 类型：Boolean 默认：true

说明：焦点离开时验证（单选/复选按钮除外）。如果表单中没有输入任何内容，所有的规则将被跳过，除非该表单已经被标记为无效的。

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

[10] onkeyup 类型：Boolean 默认：true

说明：当键盘按键弹起时验证。只要表单元素没有被标记成无效的，不会有反应。另外，所有的规则将在每次按键弹起时验证。

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

[11] onclick 类型：Boolean 默认：true

说明：鼠标点击验证针对单选和复选按钮。

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

[12] focusInvalid 类型：Boolean 默认：true

说明：当验证无效时，焦点跳到第一个无效的表单元素。当为false时，验证无效时，没有焦点响应。

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

[12] focusCleanup 类型：Boolean 默认：false

说明：如果为true，当表单得到焦点时，移除在该表单上的errorClass并隐藏所有错误消息。避免与focusInvalid一起使用。

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

[13] meta 类型：String

说明：如果想使用其它插件来使用元数据验证规则，得指定相应的元数据对象。

```
$("#myform").validate({
   meta: "validate"
})
<input type="text" name="email" class="{validate:{ required: true, email:true}}" />
```

[14] errorClass 类型：String 默认："error"

说明：用此设定的样式来定义错误消息的样式。

```
$(".selector").validate({
   errorClass: "invalid"
})
```

[15] validClass 类型：String 默认："valid"

说明：设定当验证通过时，消息显示的样式。

```
$(".selector").validate({
   validClass: "success"
})
```

[16] errorElement 类型：String 默认："label"

说明：用html元素类型创建错误消息的容器。默认的"label"有个优点就是能在错误消息与无效表单之间用for属性建立有意义的联系（一个常常使用的，而不管表单元素是什么的）。

```
$(".selector").validate({
   errorElement: "em"
})
```

[17] wrapper 类型：Boolean

说明：用一个指定的元素将错误消息包围。与errorLabelContainer一起创建一个错误消息列表非常有用。

```
$(".selector").validate({
   wrapper: "li"
})
```

[18] errorLabelContainer 类型：Selector

说明：错误消息标签的容器。

```
$("#myform").validate({
   errorLabelContainer: "#messageBox",
   wrapper: "li"
})
```

[19] errorContainer 类型：Selector

说明：错误消息的容器。

```
$("#myform").validate({
   errorContainer: "#messageBox1, #messageBox2",
   errorLabelContainer: "#messageBox1 ul",
   wrapper: "li", debug:true,
```

```
   submitHandler: function() { alert("Submitted!") }
})
```

[20] showErrors 类型：Callback 默认：None,内置的显示消息

说明：自定义消息显示的句柄。该回调函数有两个参数，第一个为errorMap,第二个参数为errorList，在validator对象的上下文中调用。参数只包含那些经过onblur/onkeyup验证的表单元素，也有可能是单个元素。除此之外，你还可以用this.defaultShowErrors()触发默认的行为。

```
$(".selector").validate({
   showErrors: function(errorMap, errorList) {
               $("#summary").html("Your form contains "
                                  + this.numberOfInvalids()
                                  + " errors, see details below.");
               this.defaultShowErrors();
        }
 })
```

[21] errorPlacement 类型：Callback 默认：紧跟在无效表单后的标签中

说明：用户自定义错误标签的显示位置。第一个参数：一个作为jQuery对象的错误标签，第二个参数为：一个作为jQuery对象的未通过验证的表单元素。

```
$("#myform").validate({
  errorPlacement: function(error, element) {
     error.appendTo( element.parent("td").next("td") );
   },
   debug:true
 })
```

[22] success 类型：String,Callback

说明：如果指定它，当验证通过时显示一个消息。如果是String类型的，则添加该样式到标签中，如果是一个回调函数，则将标签作为其唯一的参数。

```
$("#myform").validate({
   //success: "valid",
     success: function(label) {
         label.addClass("valid").text("Ok!")
     }
})
```

[23] highlight 类型：Callback 默认：添加errorClass到表单元素

说明：将未通过验证的表单元素设置高亮。

```
$(".selector").validate({
  highlight: function(element, errorClass) {
     $(element).fadeOut(function() {
       $(element).fadeIn()
     })
  }
})
```

[24] unhighlight 类型：Callback 默认：移除errorClass

说明：与highlight操作相反

```
$(".selector").validate({
  highlight: function(element, errorClass) {
     $(element).addClass(errorClass);
     $(element.form).find("label[for=" + element.id + "]")
                    .addClass(errorClass);
  },
  unhighlight: function(element, errorClass) {
     $(element).removeClass(errorClass);
     $(element.form).find("label[for=" + element.id + "]")
                    .removeClass(errorClass);
```

```
  }
});
```

[25] ignoreTitle 类型：Boolean 默认：false

说明：设置它用来跳过错误消息对title属性的引用，避免Google工具栏引起的冲突。

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

原文地址：http://docs.jquery.com/Plugins/Validation/validate#toptions

---

1.2 jQuery验证框架（二）插件方法 (jQuery validation)

发表时间: 2009-12-25

jQuery验证框架

二、插件方法(Plugin methods)

[1] validate([options]) 返回：Validator

说明：见第一部分

[2] valid() 返回：Boolean

说明：检查表单是否已通过验证。

```
$("#myform").validate();
$("a.check").click(function() {
  alert("Valid: " + $("#myform").valid());
  return false;
});
```

[3] rules() 返回：Options

说明：返回一个表单元素的验证规则。有几个方法定义验证规则：

- 在表单元素的class属性中定义验证规则（推荐的方法）。
- 通过指定验证方法的属性（推荐的方法）。
- 可以通过元数据(metadata)插件来定义元数据验证规则。
- 可以通过指定validate()方法的rules选项。

```
alert($("#password").rules()["required"]);
```

[4] rules("add", rules) 返回：Options

参数"add" 类型：String

参数 rules 类型：Options 要添加的规则，与validate方法中的验证规则一致。

说明：添加规则到匹配的表单元素，返回该元素的所有验证规则，需要先执行$("form").validate()。**在rules中也可以添加用户自定义的消息对象。**

```
$("#myinput").rules("add", {
 required: true,
 minlength: 2,
 messages: {
   required: "Required input",
   minlength: jQuery.format("Please, at least {0} characters are necessary")
 }
});
```

[5] rules("remove", [rules]) 返回：Options

参数"remove" 类型：String

参数 rules (Options) 类型：Options 用空白符分割的验证规则。只操作通过rules选项或rules("add")指定的验证规则。

说明：从第一个匹配的表单元素中移除指定的验证规则，并返回该元素所有的验证规则。

```
$("#myinput").rules("remove");
$("#myinput").rules("remove", "min max");
```

[6] removeAttrs(attributes) 返回：Options

参数 attributes 类型：String 用空白符分割的属性列表

说明：从第一个匹配的表单元素中删除指定的属性并返回它们。

```
$("#skip").click(function() {
  var rules = $("#myinput").removeAttrs("min max");
  $("#myform).submit();
  $("#myinput").attr(rules);
});
```

原文请见：http://docs.jquery.com/Plugins/

Validation#Fields_with_complex_names_.28brackets.2C_dots.29

1.3 jQuery验证框架（三、四）选择器及实用工具 (jQuery validation)

发表时间: 2009-12-26

jQuery验证框架

三、定制的选择器(Custom Selectors)

[1] :blank 返回：Array<Element>

说明：匹配所有空值的表单元素。没有任何值或都空白符(whitespace)都被认为是空值。

它是由 **jQuery.trim(value).length == 0** 来判断的。

```
$("input:blank").css("background-color", "#bbbbff");
```

[2] :filled 返回：Array<Element>

说明：匹配所有不为空的表单元素。任何值都可以认为是已输入的，但只有空白符的值除外。

它是由 **jQuery.trim(value).length > 0** 来判断的。

```
$("input:filled").css("background-color", "#bbbbff");
```

[3] :unchecked 返回：Array<Element>

说明：匹配所有未选择的表单元素。反向操作为 :checked

```
function countUnchecked() {
    var n = $("input:unchecked").length;
    $("div").text(n + (n == 1 ? " is" : " are") + " unchecked!");
}
countUnchecked();
$(":checkbox").click(countUnchecked);
```

四、实用工具(Utilities)

jQuery.validator.format(template, [argument], [argumentN...]) 返回：String

参数 template 类型：String 要格式化的字符串

参数 argument (Optional) 类型：String, Array<String> 用字符串或字符串数组(对应索引的元素)填充第一个占位符

参数 argumentN... (Optional) 类型：String 填充第二个或之后的占位符。

说明：用参数来填充{n}占位符。除template外的一个或多个参数都可以用来填充相应的占位符。

```
$(document).ready(function(){
    $("button").click(function () {
       var str = "Hello {0}, this is {1}";
       alert("'" + str + "'");
       str = jQuery.validator.format(str, "World", "Bob");
       //str = $.validator.format(str, ["koala","oo"]);
       alert("'" + str + "'");
    });
});
```

原文请见：http://docs.jquery.com/Plugins/Validation

1.4 jQuery验证框架（五）验证器 (jQuery validation)

发表时间: 2009-12-26

jQuery验证框架

五、验证器(Validator)

validate方法返回的验证器对象(Validator Object) 有一些公用的方法。你可以用来触发验证程序或改变表单(form)的内容。验证器对象有更多的方法，不过只有文档中给出的这些方法是专为使用而设计的。

(一)验证器方法(Validator methods)

[1] form() 返回：Boolean

说明：验证表单是否通过验证，若通过验证则返回true，反之返回false。这个方法在正常的提交事件(submit event)触发，它返回一个结果。

```
$("#myform").validate().form();
```

[2] element(element) 返回：Boolean

参数 element 类型：Selector 验证表单中的一个需要验证的表单元素。

说明：验证单个表单元素是否通过验证，若通过验证则返回true，反之返回false。这个方法在正常的焦点离开事件(blur)或按键弹起(keyup)时触发，它返回一个结果。

```
$("#myform").validate().element( "#myselect" );
```

[3] resetForm( ) 返回：undefined

说明：重置表单。

恢复表单元素到原来的值(需要form插件支持)，移除无效验证的样式并隐藏错误消息。

(...貌似只有在IE下才可以移除样式)

```
var validator = $("#myform").validate();
validator.resetForm();
```

[4] showErrors(errors) 返回：undefined

参数 errors 类型：Object<String, String> 一个或多个表单元素的name属性和验证消息组成的键/值对。

说明：显示指定的验证消息。

在指定的errorPlacement中显示验证消息。键为待验证表单元素的name属性，值为相应的验证消息。

```
var validator = $("#myform").validate();
validator.showErrors({"firstname": "I know that your firstname is Pete, Pete!"});
```

[5] numberOfInvalids( ) 返回：Integer

说明：返回未通过验证的表单元素的个数。

这个方法依赖于内部的验证器情况。只有在验证完所有表单元素时才统计所有待验证的表单元素(submit事件或通过$("form").valid())。当只验证单个表单元素，则只有统计该表单元素。与invalidHandler选项联合使用的时候非常有用。

```
var validator = $("#myform").validate({
    invalidHandler: function() {
        $("#summary").text(validator.numberOfInvalids() +"field(s) areinvalid");
    }
});
```

(二)验证器函数(Validator functions)

[1] setDefaults(defaults) 返回：undefined

参数 defaults 类型：Options 要设置成默认值的选项。

说明：修改验证框架的默认设置。

接受validate方法中的所有选项。

```
jQuery.validator.setDefaults({
    debug: true
});
```

[2] addMethod(name, method, [message]) 返回：undefined

参数 name 类型：String 要添加的方法名，用于标识和引用，必须是一个有效的javascript标识符。

参数 method 类型：Callback 方法的实现部分，返回true如果表单元素通过验证。

参数 message(Optional) 类型：String, Function 该方法的默认验证消息。可以用 **jQuery.validator.format(value)** 方法创建。如果未定义该参数，则使用本地已存在的验证消息，另外，必须为指定的表单元素定义验证消息。

说明：添加一个用户自定义的验证方法。它由方法名（必须是一个合法的javascript标识符）、基于javascript的函数及默认的验证消息组成。

```
jQuery.validator.addMethod("math", function(value, element, params) {
    return this.optional(element) || value == params[0] + params[1];
}, jQuery.format("Please enter the correct value for {0} + {1}"));
```

[3] addClassRules(name, rules) 返回：undefined

参数 name 类型：String 要添加的样式规则名。

参数 rules 类型：Options 规则选项。

说明：添加一个复合的样式验证方法。对于将多个联合使用的规则重构进单个样式中非常有用。

```
jQuery.validator.addClassRules("name", {
  required: true,
  minlength: 2
});
```

[4] addClassRules(rules) 返回：undefined

参数 rules 类型：Options 样式类名-规则表。

说明：添加一个复合的样式验证方法。对于重构通用的联合规则非常有用。

```
jQuery.validator.addClassRules({
  name: {
    required: true,
    minlength: 2
  },
  zip: {
    required: true,
    digits: true,
    minlength: 5,
    maxlength: 5
  }
});
```

原文请见：http://docs.jquery.com/Plugins/Validation

1.5 jQuery验证框架（六）内置验证方法 (jQuery validation)

发表时间: 2009-12-26

jQuery验证框架

六、框架内建的验证方法(List of built-in Validation methods)

[1] required( ) 返回：Boolean

说明：让表单元素必须填写（选择）。

如果表单元素为空(text input)或未选择(radio/checkbox)或选择了一个空值(select)。

作用于text inputs, selects, checkboxes and radio buttons.

当select提供了一个空值选项<option value="">Choose...</option>则强迫用户去选择一个不为空的值。

```
$("#myform").validate({
  rules: {
    fruit: "required"
  }
});
```

[2] required(dependency-expression) 返回：Boolean

参数 dependency-expression 类型：String 在form上下文中的一个表达式(String)，表单元素是否需要填写依赖于该表达式返回一个或多个元素。

说明：让表单元素必须填写（选择），依赖于参数的返回值。

表达式中像#foo:checked, #foo:filled, #foo:visible这样的选择过滤器将经常用到。

```
$("#myform").validate({
  rules: {
    details: {
      required: "#other:checked"
    }
  }, debug:true
});
$("#other").click(function() {
   $("#details").valid();
});
```

[3] required(dependency-callback) 返回：Boolean

参数 dependency-callback 类型：Callback 该回函数以待验证表单元素作为其唯一的参数。当该回调函数返回true，则该表单元素是必须的。

说明：让表单元素必须填写（选择），依赖于参数的返回值。

表达式中像#foo:checked, #foo:filled, #foo:visible这样的选择过滤器将经常用到。

```
$("#myform").validate({
  rules: {
    age: {
      required: true,
      min: 3
    },
    parent: {
      required: function(element) {
        return $("#age").val() < 13;
      }
    }
  }
});
$("#age").blur(function() {
    $("#parent").valid();
});
```

[4] remote(options) 返回：Boolean

参数 options 类型：String, Options 请求服务器端资源的url(String)。或$.ajax()方法中的选项(Options)。

说明：请求服务器端资源验证。

服务器端的资源通过$.ajax (XMLHttpRequest)获取key/value对，响应返回true则表单通过验证。

```
$("#myform").validate({
  rules: {
    email: {
      required: true,
      email: true,
      remote: "check-email.php"
    }
```

```
  }
});
```

[5] minlength(length) 返回：Boolean

参数 length 类型：Integer 至少需要多少个字符数。

说明：确保表单元素满足给定的最小字符数。

在文本框(text input)中输入的字符太少、没有选中足够的复选框(checkbox)、一个选择框(select)中没有选中足够的选项。这以上三种情况中该方法返回false。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      minlength: 3
    }
  }
});
```

[6] maxlength(length) 返回：Boolean

参数 length 类型：Integer 允许输入的最大字符数。

说明：确保表单元素的文本不超过给定的最大字符数。

在文本框(text input)中输入的字符太多、选择太多的复选框(checkbox)、一个选择框(select)中没有选中太多的选项。这以上三种情况中该方法返回false。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      maxlength: 4
    }
  }
});
```

[7] rangelength(range) 返回：Boolean

参数 range 类型：Array<Integer> 允许输入的字符数范围。

说明：确保表单元素的文本字符数在给定的范围当中。

在文本框(text input)中输入的字符数不在给定范围内、选择的复选框(checkbox)不在给在的范围内、一个选择框(select)选中的选项不在给定的范围内。这以上三种情况中该方法返回false。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      rangelength: [2, 6]
    }
  }
});
```

[8] min(value) 返回：Boolean

参数 value 类型：Integer 需要输入的最小整数。

说明：确保表单元素的值大于等于给定的最小整数。

该方法只在文本输入框(text input)下有效。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      min: 13
    }
  }
});
```

[9] max(value) 返回：Boolean

参数 value 类型：Integer 给定的最大整数。

说明：确保表单元素的值小于等于给定的最大整数。

该方法只在文本输入框(text input)下有效。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
```

```
      max: 23
    }
  }
});
```

[10] range(range) 返回：Boolean

参数 range 类型：Array<Integer> 给定的整数范围。

说明：确保表单元素的值在给定的范围当中。

该方法只在文本输入框(text input)下有效。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      range: [13, 23]
    }
  }
});
```

[11] email( ) 返回：Boolean

说明：确保表单元素的值为一个有效的email地址。

如果值为一个有效的email地址，则返回true。该方法只在文本输入框(text input)下有效。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      email: true
    }
  }
});
```

[12] url( ) 返回：Boolean

说明：确保表单元素的值为一个有效的URL地址(http://www.mydomain.com)。

如果值为一个有效的url地址，则返回true。该方法只在文本输入框(text input)下有效。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      url: true
    }
  }
});
```

[13] date( ) dateISO( ) dateDE( ) 返回：Boolean

说明：用来验证有效的日期。这三个函数分别验证的日期格式为(mm/dd/yyyy)、(yyyy-mm-dd,yyyy/mm/dd)、(mm.dd.yyyy)。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      date: true
      /*dateISO: true
        dateDE: true*/
    }
  }
});
```

[14] number( ) numberDE() 返回：Boolean

说明：用来验证小数。number()的小数点为圆点(.)，numberDE()的小数点为英文逗号(,)。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      number: true
      //numberDE: true
    }
```

```
  }
});
```

[15] digits() 返回：Boolean

说明：确保文本框中的值为数字。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      digits: true
    }
  }
});
```

[16] digits() 返回：Boolean

说明：确保文本框中的值为数字。

```
$("#myform").validate({
  rules: {
    field: {
      required: true,
      digits: true
    }
  }
});
```

[17] accept([extension]) 返回：Boolean

参数 extension(Optional) 类型：String 允许的文件后缀名，用"|"或","分割。默认为"png|jpe?g|gif"

说明：确保表单元素接收给定的文件后缀名的文件。如果没有指定参数，则只有图片是允许的(png,jpeg,gif)。

```
$("#myform").validate({
  rules: {
    field: {
```

```
      required: true,
      accept: "xls|csv"
    }
  }
});
```

[18] equalTo(other) 返回：Boolean

参数 other 类型：Selector 要与当前值比较的另一个表单元素。

说明：确保两个表单元素的值是一致的。

```
$("#myform").validate({
  rules: {
    password: "required",
    password_again: {
      equalTo: "#password"
    }
  }
});
```

原文请见：http://docs.jquery.com/Plugins/Validation

1.6 jQuery验证框架（七）注意事项 (jQuery validation)

发表时间: 2009-12-29

jQuery验证框架

七、注意事项

[1]复杂的name属性值

当使用rules选项时，如果表单的name属性值包含有**非法的javascript标识符**，必须将name值加上引号。

```
$("#myform").validate({
  rules: {
    // no quoting necessary
    name: "required",
    // quoting necessary!
    "user[email]": "email",
    // dots need quoting, too!
    "user.address.street": "required"
  }
});
```

[2]重构规则

不论什么时候，当你的表单中的多个字段含有相同的验证规则及验证消息，重构规则可以减少很多重复。使用 **addMethod** 和 **addClassRules** 将非常有效果。

假使已经重构了如下规则：

```
// alias required to cRequired with new message
$.validator.addMethod("cRequired", $.validator.methods.required,
  "Customer name required");
// alias minlength, too
$.validator.addMethod("cMinlength", $.validator.methods.minlength,
  // leverage parameter replacement for minlength, {0} gets replaced with 2
  $.format("Customer name must have at least {0} characters"));
// combine them both, including the parameter for minlength
$.validator.addClassRules("customer", { cRequired: true, cMinlength: 2 });
```

那么使用的时候如下：

```
<input name="customer1" class="customer" />
<input name="customer2" class="customer" />
<input name="customer3" class="customer" />
```

[3]验证消息

当验证了一个无效的表单元素，验证消息显示在用户面前。这些消息是从哪里来的呢？有三个途径来取得验证消息。

1.通过待验证表单元素的title属性

2.通过默认的验证消息

3.通过插件设置(messages选项)

这三种途径的优先顺序为：**3 > 1 > 2**

[4]验证消息与Google工具栏的冲突

有时候验证消息会与Goole工具栏的AutoFill插件冲突。AutoFill通过替换表单元素的title属性，以显示提示消息。此时，验证消息如果获取的是title属性值，那么就得不到我们预期想要得到的结果。当文档载入时，可以通过如下方法避免冲突。

```
$("input.remove_title").attr("title", "");
```

[5]表单提交

默认地，表单验证失败时阻止表单的提交，当验证通过，表单提交。当然，也可以通过**submitHandler**来自定义提交事件。

将提交按钮的class属性设置成cancel，在表单提交时可以跳过验证。

```
<input type="submit" name="submit" value="Submit" />
<input type="submit" class="cancel" name="cancel" value="Cancel" />
```

下面这段代码将循环提交表单：

```
$("#myform").validate({
 submitHandler: function(form) {
```

```
   // some other code maybe disabling submit button
   // then:
   $(form).submit();
 }
});
```

$(form).submit() 触发了另外一轮的验证，验证后又去调用submitHandler，然后就循环了。可以用form.submit() 来触发原生的表单提交事件。

```
$("#myform").validate({
 submitHandler: function(form) {
   form.submit();
 }
});
```

原文请见：http://docs.jquery.com/Plugins/Validation

1.7 jQuery验证框架（八）应用实例 (jQuery validation)

发表时间: 2009-12-29

jQuery验证框架

八、应用实例

[1] 验证页面

```
<%@ page language="java" import="java.util.*" pageEncoding="gb2312"%>
<html>
  <head>
    <title>jquery验证框架</title>
    <link rel="stylesheet" type="text/css" href="css/index.css">
    <script type="text/javascript" src=js/jquery-1.3.2.min.js></script>
    <script type="text/javascript" src=js/jquery.validate.pack.js></script>
    <script type="text/javascript" src=js/jquery.form.js></script>
    <script type="text/javascript" src=js/valid.js></script>
    <style type="text/css">
            label { width: 10em; float: left; }
                label.haha {color: red; padding-left: 18px; vertical-align: top;width: 196px; b
                input.haha { border: 1px solid red; }
                label.valid {background: url("images/checked.gif") no-repeat; color: #065FB9;}
                input.focus { border: 2px solid green; }
                ul li{ display: block;}
    </style>
  </head>

  <body>

  <div id="form_con">
                <form class="cmxform" id="myform" method="post" action="">
                        <table cellspacing="0" cellpadding="0">
                                <tbody>
                                        <tr>
                                                <td>用户名</td>
                                                <td><input type="text" name="username" class="r
```

```
                <td></td>
            </tr>
            <tr>
                <td>密码</td>
                <td><input id="password" type="password" name="
                <td></td>
            </tr>
            <tr>
                <td>验证密码</td>
                <td><input type="password" name="secondpwd" /><
                <td></td>
            </tr>
            <tr>
                <td>性别</td>
                <td><input id="sex" type="radio" name="sex" />男
                <td></td>
            </tr>
            <tr>
                <td>年龄</td>
                <td><input type="text" name="age" /></td>
                <td></td>
            </tr>
            <tr>
                <td>邮箱</td>
                <td><input type="text" name="email" /></td>
                <td></td>
            </tr>
            <tr>
                <td>个人网页</td>
                <td><input type="text" name="purl" /></td>
                <td></td>
            </tr>
            <tr>
                <td>电话</td>
                <td><input type="text" name="telephone" /></td>
                <td></td>
            </tr>
```

```
                                        <tr>
                                                <td>附件</td>
                                                <td><input type="file" name="afile"/></td>
                                                <td></td>
                                        </tr>
                                        <tr><td colspan="3" ><input type="submit" name="submit'
                                </tbody>
                        </table>
                </form>
  </div>
  </body>
</html>
```

[1] 验证js

```
$(function(){
        var validator = $("#myform").validate({
                debug: true,       //调试模式取消submit的默认提交功能
                errorClass: "haha",//默认为错误的样式类为：error
                focusInvalid: false,
                onkeyup: false,
                submitHandler: function(form){   //表单提交句柄,为一回调函数，带一个参数：form
                        alert("提交表单");
                        //form.submit();   提交表单
                },
                rules: {           //定义验证规则,其中属性名为表单的name属性
                        username: {
                                required: true,
                                minlength: 2,
                                remote: "uservalid.jsp"  //传说当中的ajax验证
                        },
                        firstpwd: {
                                required: true,
                                //minlength: 6
                                rangelength: [6,8]
                        },
```

```
                secondpwd: {
                        required: true,
                        equalTo: "#password"
                },
                sex: {
                        required: true
                },
                age: {
                        required: true,
                        range: [0,120]
                },
                email: {
                        required: true,
                        email: true
                },
                purl: {
                        required: true,
                        url: true
                },
                afile: {
                        required: true,
                        accept: "xls,doc,rar,zip"
                }
        },
        messages: {       //自定义验证消息
                username: {
                        required: "用户名是必需的！",
                        minlength: $.format("用户名至少要{0}个字符！"),
                        remote: $.format("{0}已经被占用")
                },
                firstpwd: {
                        required: "密码是必需的！",
                        rangelength: $.format("密码要在{0}-{1}个字符之间！")
                },
                secondpwd: {
                        required: "密码验证是必需的！",
                        equalTo: "密码验证需要与密码一致"
```

```
            },
            sex: {
                    required: "性别是必需的"
            },
            age: {
                    required: "年龄是必需的",
                    range: "年龄必须介于{0}-{1}之间"
            },
            email: {
                    required: "邮箱是必需的！",
                    email: "请输入正确的邮箱地址（例如 myemail@163.com）"
            },
            purl: {
                    required: "个人主页是必需的",
                    url: "请输入正确的url格式,如 http://www.domainname.com"
            },
            afile: {
                    required: "附件是必需的！",
                    accept: "只接收xls,doc,rar,zip文件"
            }
    },
    errorPlacement: function(error, element) {  //验证消息放置的地方
    error.appendTo( element.parent("td").next("td") );
},
highlight: function(element, errorClass) {  //针对验证的表单设置高亮
        $(element).addClass(errorClass);
},
success: function(label) {
        label.addClass("valid").text("Ok!")
            }

/*,
errorContainer: "#error_con",              //验证消息集中放置的容器
errorLabelContainer: "#error_con ul",      //存放消息无序列表的容器
wrapper: "li",                                                    //将验证
validClass: "valid",                                            //通过验证的样式
errorElement: "em",                                             //验证标签的名称
```

```
            success: "valid"                                                    //验证通过的样式
            */
        });
        $("button").click(function(){
                validator.resetForm();
        });
        //alert($("#password").rules()["required"]);
        //validator.showErrors({"username": "用户名是必需的"});
        /*$("button").click(function () {
            var str = "Hello {0}, this is {1}";
            alert("'" + str + "'");
            str = $.validator.format(str, ["koala","oo"]);
            alert("'" + str + "'");
    });*/

});
```

[3] 远程验证程序

```
<%@ page language="java" import="java.io.PrintWriter" pageEncoding="gb2312"%><%
        String username = request.getParameter("username");
        PrintWriter pw = response.getWriter();
        try{
                if(username.toLowerCase().equals("admin")){
                        pw.println("true");
                }else{
                        pw.println("false");
                }
        }catch(Exception ex){
                ex.getStackTrace();
        }finally{
                pw.flush();
                pw.close();
        }
%>
```

[4] 验证效果

见附件

--