jQuery validation

**jQuery验证框架**

Html代码

1. <script type="text/javascript" src=js/jquery-1.3.2.min.js></script>
2. <script type="text/javascript" src=js/jquery.validate.pack.js></script>
3. <script type="text/javascript">
4. $(document).ready(function(){
5. $("#textForm").validate();
6. });
7. </script>
8. <form class="cmxform" id="commentForm" method="get" action="">
9. Name<input id="cname" name="name" size="25" class="required" minlength="2" />
10. <input class="submit" type="submit" value="Submit"/>
11. </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。

Js代码

1. $(".selector").validate({
2. debug: true
3. })

**[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

Js代码

1. $(".selector").validate({
2. submitHandler: function(form) {
3. // do other stuff for a valid form
4. form.submit();
5. }
6. })

**[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）

Js代码

1. $(".selector").validate({
2. invalidHandler: function(form, validator) {
3. var errors = validator.numberOfInvalids();
4. if (errors) {
5. var message = errors == 1
6. ? 'You missed 1 field. It has been highlighted'
7. : 'You missed ' + errors + ' fields. They have been highlighted';
8. $("div.error span").html(message);
9. $("div.error").show();
10. } else {
11. $("div.error").hide();
12. }
13. }
14. })

**[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。

Js代码

1. $("#myform").validate({
2. ignore: ".ignore"
3. })

**[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。

Js代码

1. $(".selector").validate({
2. rules: {
3. // simple rule, converted to {required:true}
4. name: "required",
5. // compound rule
6. email: {
7. required: true,
8. email: true
9. }/\*
10. email: {
11. depends: function(element) {
12. return $("#contactform\_email:checked")
13. }
14. }\*/
15. }
16. })

**[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。

Js代码

1. $(".selector").validate({
2. rules: {
3. name: "required",
4. email: {
5. required: true,
6. email: true
7. }
8. },
9. messages: {
10. name: "Please specify your name",
11. email: {
12. required: "We need your email address to contact you",
13. email: "Your email address must be in the format of name@domain.com"
14. }
15. }
16. })

**[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。

Js代码

1. $("#myform").validate({
2. groups: {
3. username: "fname lname"
4. },
5. errorPlacement: function(error, element) {
6. if (element.attr("name") == "fname"
7. || element.attr("name") == "lname" )
8. error.insertAfter("#lastname");
9. else
10. error.insertAfter(element);
11. },
12. debug:true
13. })

**[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。

Js代码

1. $(".selector").validate({
2. onsubmit: false
3. })

**[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。

Js代码

1. $(".selector").validate({
2. onfocusout: false
3. })

**[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。

Js代码

1. $(".selector").validate({
2. onkeyup: false
3. })

**[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。

Js代码

1. $(".selector").validate({
2. onclick: false
3. })

**[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。

Js代码

1. $(".selector").validate({
2. focusInvalid: false
3. })

**[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。

Js代码

1. $(".selector").validate({
2. focusCleanup: true
3. })

**[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。

Js代码

1. $("#myform").validate({
2. meta: "validate"
3. })
4. <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。

Js代码

1. $(".selector").validate({
2. errorClass: "invalid"
3. })

**[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。

Js代码

1. $(".selector").validate({
2. validClass: "success"
3. })

**[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）。

Js代码

1. $(".selector").validate({
2. errorElement: "em"
3. })

**[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。

Js代码

1. $(".selector").validate({
2. wrapper: "li"
3. })

**[18]  errorLabelContainer**      类型：Selector   
      说明：错误消息标签的容器。

Js代码

1. $("#myform").validate({
2. errorLabelContainer: "#messageBox",
3. wrapper: "li"
4. })

**[19]  errorContainer**      类型：Selector   
      说明：错误消息的容器。

Js代码

1. $("#myform").validate({
2. errorContainer: "#messageBox1, #messageBox2",
3. errorLabelContainer: "#messageBox1 ul",
4. wrapper: "li", debug:true,
5. submitHandler: function() { alert("Submitted!") }
6. })

**[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。

Js代码

1. $(".selector").validate({
2. showErrors: function(errorMap, errorList) {
3. $("#summary").html("Your form contains "
4. + this.numberOfInvalids()
5. + " errors, see details below.");
6. this.defaultShowErrors();
7. }
8. })

**[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。

Js代码

1. $("#myform").validate({
2. errorPlacement: function(error, element) {
3. error.appendTo( element.parent("td").next("td") );
4. },
5. debug:true
6. })

**[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。

Js代码

1. $("#myform").validate({
2. //success: "valid",
3. success: function(label) {
4. label.addClass("valid").text("Ok!")
5. }
6. })

**[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。

Js代码

1. $(".selector").validate({
2. highlight: function(element, errorClass) {
3. $(element).fadeOut(function() {
4. $(element).fadeIn()
5. })
6. }
7. })

**[24]  unhighlight**      类型：Callback    默认：移除errorClass   
      说明：与highlight操作相反

Js代码

1. $(".selector").validate({
2. highlight: function(element, errorClass) {
3. $(element).addClass(errorClass);
4. $(element.form).find("label[for=" + element.id + "]")
5. .addClass(errorClass);
6. },
7. unhighlight: function(element, errorClass) {
8. $(element).removeClass(errorClass);
9. $(element.form).find("label[for=" + element.id + "]")
10. .removeClass(errorClass);
11. }
12. });

**[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。

Js代码

1. $(".selector").validate({
2. ignoreTitle: true
3. })

**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。

Js代码

1. $("#myform").validate();
2. $("a.check").click(function() {
3. alert("Valid: " + $("#myform").valid());
4. return false;
5. });

$("#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。

Js代码

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

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。**

Js代码

1. $("#myinput").rules("add", {
2. required: true,
3. minlength: 2,
4. messages: {
5. required: "Required input",
6. minlength: jQuery.format("Please, at least {0} characters are necessary")
7. }
8. });

$("#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。

Js代码

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

$("#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。

Js代码

1. $("#skip").click(function() {
2. var rules = $("#myinput").removeAttrs("min max");
3. $("#myform).submit();
4. $("#myinput").attr(rules);
5. });

**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。

Js代码

1. $("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。

Js代码

1. $("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

Js代码

1. function countUnchecked() {
2. var n = $("input:unchecked").length;
3. $("div").text(n + (n == 1 ? " is" : " are") + " unchecked!");
4. }
5. countUnchecked();
6. $(":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。

Js代码

1. $(document).ready(function(){
2. $("button").click(function () {
3. var str = "Hello {0}, this is {1}";
4. alert("'" + str + "'");
5. str = jQuery.validator.format(str, "World", "Bob");
6. //str = $.validator.format(str, ["koala","oo"]);
7. alert("'" + str + "'");
8. });
9. });

**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。

Js代码

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

$("#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。

Js代码

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

$("#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)

Js代码

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

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。

Js代码

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

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。

Js代码

1. var validator = $("#myform").validate({
2. invalidHandler: function() {
3. $("#summary").text(validator.numberOfInvalids() +"field(s) areinvalid");
4. }
5. });

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。

Js代码

1. jQuery.validator.setDefaults({
2. debug: true
3. });

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。

Js代码

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

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。

Js代码

1. jQuery.validator.addClassRules("name", {
2. required: true,
3. minlength: 2
4. });

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。

Js代码

1. jQuery.validator.addClassRules({
2. name: {
3. required: true,
4. minlength: 2
5. },
6. zip: {
7. required: true,
8. digits: true,
9. minlength: 5,
10. maxlength: 5
11. }
12. });

**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。

Js代码

1. $("#myform").validate({
2. rules: {
3. fruit: "required"
4. }
5. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. details: {
4. required: "#other:checked"
5. }
6. }, debug:true
7. });
8. $("#other").click(function() {
9. $("#details").valid();
10. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. age: {
4. required: true,
5. min: 3
6. },
7. parent: {
8. required: function(element) {
9. return $("#age").val() < 13;
10. }
11. }
12. }
13. });
14. $("#age").blur(function() {
15. $("#parent").valid();
16. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. email: {
4. required: true,
5. email: true,
6. remote: "check-email.php"
7. }
8. }
9. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. minlength: 3
6. }
7. }
8. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. maxlength: 4
6. }
7. }
8. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. rangelength: [2, 6]
6. }
7. }
8. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. min: 13
6. }
7. }
8. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. max: 23
6. }
7. }
8. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. range: [13, 23]
6. }
7. }
8. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. email: true
6. }
7. }
8. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. url: true
6. }
7. }
8. });

**[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)。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. date: true
6. /\*dateISO: true
7. dateDE: true\*/
8. }
9. }
10. });

**[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( , )。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. number: true
6. //numberDE: true
7. }
8. }
9. });

**[15]  digits()**      返回：Boolean   
      说明：确保文本框中的值为数字。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. digits: true
6. }
7. }
8. });

**[16]  digits()**      返回：Boolean   
      说明：确保文本框中的值为数字。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. digits: true
6. }
7. }
8. });

**[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)。

Js代码

1. $("#myform").validate({
2. rules: {
3. field: {
4. required: true,
5. accept: "xls|csv"
6. }
7. }
8. });

**[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。

Js代码

1. $("#myform").validate({
2. rules: {
3. password: "required",
4. password\_again: {
5. equalTo: "#password"
6. }
7. }
8. });

**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。

Js代码

1. $("#myform").validate({
2. rules: {
3. // no quoting necessary
4. name: "required",
5. // quoting necessary!
6. "user[email]": "email",
7. // dots need quoting, too!
8. "user.address.street": "required"
9. }
10. });

**[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：

Js代码

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

    那么使用的时候如下：

Html代码

1. <input name="customer1" class="customer" />
2. <input name="customer2" class="customer" />
3. <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。

Js代码

1. $("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。

Js代码

1. <input type="submit" name="submit" value="Submit" />
2. <input type="submit" class="cancel" name="cancel" value="Cancel" />

    下面这段代码将循环提交表单：

Java代码

1. $("#myform").validate({
2. submitHandler: function(form) {
3. // some other code maybe disabling submit button
4. // then:
5. $(form).submit();
6. }
7. });

    $(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。

Java代码

1. $("#myform").validate({
2. submitHandler: function(form) {
3. form.submit();
4. }
5. });

**八、应用实例**   
  
**[1] 验证页面**

Html代码

1. <%@ page language="java" import="java.util.\*" pageEncoding="gb2312"%>
2. <html>
3. <head>
4. <title>jquery验证框架</title>
5. <link rel="stylesheet" type="text/css" href="css/index.css">
6. <script type="text/javascript" src=js/jquery-1.3.2.min.js></script>
7. <script type="text/javascript" src=js/jquery.validate.pack.js></script>
8. <script type="text/javascript" src=js/jquery.form.js></script>
9. <script type="text/javascript" src=js/valid.js></script>
10. <style type="text/css">
11. label { width: 10em; float: left; }
12. label.haha {color: red; padding-left: 18px; vertical-align: top;width: 196px; background: url("images/unchecked.gif") no-repeat;}
13. input.haha { border: 1px solid red; }
14. label.valid {background: url("images/checked.gif") no-repeat; color: #065FB9;}
15. input.focus { border: 2px solid green; }
16. ul li{ display: block;}
17. </style>
18. </head>
20. <body>
22. <div id="form\_con">
23. <form class="cmxform" id="myform" method="post" action="">
24. <table cellspacing="0" cellpadding="0">
25. <tbody>
26. <tr>
27. <td>用户名</td>
28. <td><input type="text" name="username" class="required" /></td>
29. <td></td>
30. </tr>
31. <tr>
32. <td>密码</td>
33. <td><input id="password" type="password" name="firstpwd" /></td>
34. <td></td>
35. </tr>
36. <tr>
37. <td>验证密码</td>
38. <td><input type="password" name="secondpwd" /></td>
39. <td></td>
40. </tr>
41. <tr>
42. <td>性别</td>
43. <td><input id="sex" type="radio" name="sex" />男<input type="radio" name="sex" />女</td>
44. <td></td>
45. </tr>
46. <tr>
47. <td>年龄</td>
48. <td><input type="text" name="age" /></td>
49. <td></td>
50. </tr>
51. <tr>
52. <td>邮箱</td>
53. <td><input type="text" name="email" /></td>
54. <td></td>
55. </tr>
56. <tr>
57. <td>个人网页</td>
58. <td><input type="text" name="purl" /></td>
59. <td></td>
60. </tr>
61. <tr>
62. <td>电话</td>
63. <td><input type="text" name="telephone" /></td>
64. <td></td>
65. </tr>
66. <tr>
67. <td>附件</td>
68. <td><input type="file" name="afile"/></td>
69. <td></td>
70. </tr>
71. <tr><td colspan="3" ><input type="submit" name="submit" value="提交" /><button>重置</button></td></tr>
72. </tbody>
73. </table>
74. </form>
75. </div>
76. </body>
77. </html>

**[2] 验证js**

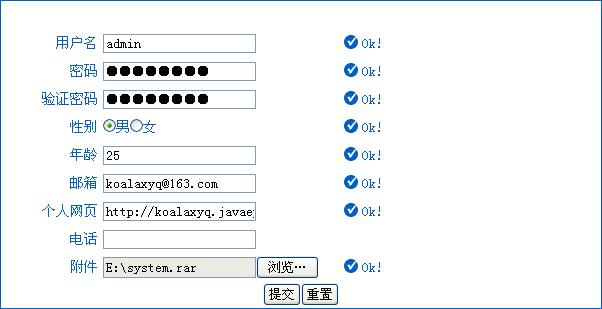
Js代码

1. $(function(){
2. var validator = $("#myform").validate({
3. debug: true,       //调试模式取消submit的默认提交功能
4. errorClass: "haha",//默认为错误的样式类为：error
5. focusInvalid: false,
6. onkeyup: false,
7. submitHandler: function(form){   //表单提交句柄,为一回调函数，带一个参数：form
8. alert("提交表单");
9. //form.submit();   提交表单
10. },
11. rules: {           //定义验证规则,其中属性名为表单的name属性
12. username: {
13. required: true,
14. minlength: 2,
15. remote: "uservalid.jsp"  //传说当中的ajax验证
16. },
17. firstpwd: {
18. required: true,
19. //minlength: 6
20. rangelength: [6,8]
21. },
22. secondpwd: {
23. required: true,
24. equalTo: "#password"
25. },
26. sex: {
27. required: true
28. },
29. age: {
30. required: true,
31. range: [0,120]
32. },
33. email: {
34. required: true,
35. email: true
36. },
37. purl: {
38. required: true,
39. url: true
40. },
41. afile: {
42. required: true,
43. accept: "xls,doc,rar,zip"
44. }
45. },
46. messages: {       //自定义验证消息
47. username: {
48. required: "用户名是必需的！",
49. minlength: $.format("用户名至少要{0}个字符！"),
50. remote: $.format("{0}已经被占用")
51. },
52. firstpwd: {
53. required: "密码是必需的！",
54. rangelength: $.format("密码要在{0}-{1}个字符之间！")
55. },
56. secondpwd: {
57. required: "密码验证是必需的！",
58. equalTo: "密码验证需要与密码一致"
59. },
60. sex: {
61. required: "性别是必需的"
62. },
63. age: {
64. required: "年龄是必需的",
65. range: "年龄必须介于{0}-{1}之间"
66. },
67. email: {
68. required: "邮箱是必需的！",
69. email: "请输入正确的邮箱地址（例如 myemail@163.com）"
70. },
71. purl: {
72. required: "个人主页是必需的",
73. url: "请输入正确的url格式,如 http://www.domainname.com"
74. },
75. afile: {
76. required: "附件是必需的！",
77. accept: "只接收xls,doc,rar,zip文件"
78. }
79. },
80. errorPlacement: function(error, element) {  //验证消息放置的地方
81. error.appendTo( element.parent("td").next("td") );
82. },
83. highlight: function(element, errorClass) {  //针对验证的表单设置高亮
84. $(element).addClass(errorClass);
85. },
86. success: function(label) {
87. label.addClass("valid").text("Ok!")
88. }
90. /\*,
91. errorContainer: "#error\_con",               //验证消息集中放置的容器
92. errorLabelContainer: "#error\_con ul",       //存放消息无序列表的容器
93. wrapper: "li",                              //将验证消息用无序列表包围
94. validClass: "valid",                        //通过验证的样式类
95. errorElement: "em",                         //验证标签的名称，默认为：label
96. success: "valid"                            //验证通过的样式类
97. \*/
98. });
99. $("button").click(function(){
100. validator.resetForm();
101. });
102. //alert($("#password").rules()["required"]);
103. //validator.showErrors({"username": "用户名是必需的"});
104. /\*$("button").click(function () {
105. var str = "Hello {0}, this is {1}";
106. alert("'" + str + "'");
107. str = $.validator.format(str, ["koala","oo"]);
108. alert("'" + str + "'");
109. });\*/
111. });

**[3] 远程验证程序**

Java代码

1. <%@ page language="java" import="java.io.PrintWriter" pageEncoding="gb2312"%><%
2. String username = request.getParameter("username");
3. PrintWriter pw = response.getWriter();
4. try{
5. if(username.toLowerCase().equals("admin")){
6. pw.println("true");
7. }else{
8. pw.println("false");
9. }
10. }catch(Exception ex){
11. ex.getStackTrace();
12. }finally{
13. pw.flush();
14. pw.close();
15. }
16. %>

**[4] 验证效果**   
  
[](http://dl.javaeye.com/upload/attachment/186738/12054387-758f-37da-a5ec-a106e270fbae.jpeg)   
  
[](http://dl.javaeye.com/upload/attachment/186740/cfe5e937-2854-3ad4-a9ee-b99b3d632815.jpeg)   
  
[](http://dl.javaeye.com/upload/attachment/186742/266a0183-b9bc-3f85-8913-99dbdab98177.jpeg)

# 九 jquery validate.addMethod 正则表达式 (自定义验证方法)

<script type="text/javascript">  
        jQuery.validator.addMethod("regex",  //addMethod第1个参数:方法名称  
        function(value, element, params) {     //addMethod第2个参数:验证方法，参数（被验证元素的值，被验证元素，参数）  
            var exp = new RegExp(params);     //实例化正则对象，参数为传入的正则表达式  
            return exp.test(value);                    //测试是否匹配  
        },  
        "格式错误");    //addMethod第3个参数:默认错误信息  
        $(function() {  
            $("#signupForm").validate(  
  
            {  
            rules: {  
                txtPassword1: "required",  //密码1必填  
  
                txtPassword2: {    //密码2的描述多于1项使用对象类型  
                    required: true,  //必填，这里可以是一个匿名方法  
                    equalTo: "#txtPassword1",  //必须与密码1相等  
                    rangelength: [5, 10],    //长度5-10之间  
                    regex: "^\\w+$"          //使用自定义的验证规则,在上例中新增的部分  
                },  
  
                txtEmail: "email"   //电子邮箱必须合法  
            },  
            messages: {  
                txtPassword1: "您必须填写",  
  
                txtPassword2: {  
                    required: "您必须填写",  
                    equalTo: "密码不一致",  
                    rangelength: "长度必须介于{0} 和 {1} 之间的字符串",  
                    regex: "密码只能是数字、字母与下划线"  
                }  
            },  
            debug: false,  //如果修改为true则表单不会提交  
            submitHandler: function() {  
                alert("开始提交了");  
            }  
        });  
    });  
    </script>

注意,如果参数有多个可以使用数组，如regex:[1,3,5],方法中可以使用下标访问：params[0]

网友提供的一些扩展验证方法:  
// 手机号码验证  
jQuery.validator.addMethod("mobile", function(value, element) {  
    var length = value.length;  
    var mobile =  /^(((13[0-9]{1})|(15[0-9]{1}))+\d{8})$/  
    return this.optional(element) || (length == 11 && mobile.test(value));  
}, "手机号码格式错误");

// 电话号码验证     
jQuery.validator.addMethod("phone", function(value, element) {  
    var tel = /^(0[0-9]{2,3}\-)?([2-9][0-9]{6,7})+(\-[0-9]{1,4})?$/;  
    return this.optional(element) || (tel.test(value));  
}, "电话号码格式错误");

// 邮政编码验证     
jQuery.validator.addMethod("zipCode", function(value, element) {  
    var tel = /^[0-9]{6}$/;  
    return this.optional(element) || (tel.test(value));  
}, "邮政编码格式错误");

// QQ号码验证     
jQuery.validator.addMethod("qq", function(value, element) {  
    var tel = /^[1-9]\d{4,9}$/;  
    return this.optional(element) || (tel.test(value));  
}, "qq号码格式错误");

// IP地址验证  
jQuery.validator.addMethod("ip", function(value, element) {  
    var ip = /^(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$/;  
    return this.optional(element) || (ip.test(value) && (RegExp.$1 < 256 && RegExp.$2 < 256 && RegExp.$3 < 256 && RegExp.$4 < 256));  
}, "Ip地址格式错误");

// 字母和数字的验证  
jQuery.validator.addMethod("chrnum", function(value, element) {  
    var chrnum = /^([a-zA-Z0-9]+)$/;  
    return this.optional(element) || (chrnum.test(value));  
}, "只能输入数字和字母(字符A-Z, a-z, 0-9)");

// 中文的验证  
jQuery.validator.addMethod("chinese", function(value, element) {  
    var chinese = /^[\u4e00-\u9fa5]+$/;  
    return this.optional(element) || (chinese.test(value));  
}, "只能输入中文");

// 下拉框验证  
$.validator.addMethod("selectNone", function(value, element) {  
    return value == "请选择";  
}, "必须选择一项");

// 字节长度验证  
jQuery.validator.addMethod("byteRangeLength", function(value, element, param) {  
    var length = value.length;  
    for (var i = 0; i < value.length; i++) {  
        if (value.charCodeAt(i) > 127) {  
            length++;  
        }  
    }  
    return this.optional(element) || (length >= param[0] && length <= param[1]);  
}, $.validator.format("请确保输入的值在{0}-{1}个字节之间(一个中文字算2个字节)"));