第 5 章　表 单 验 证

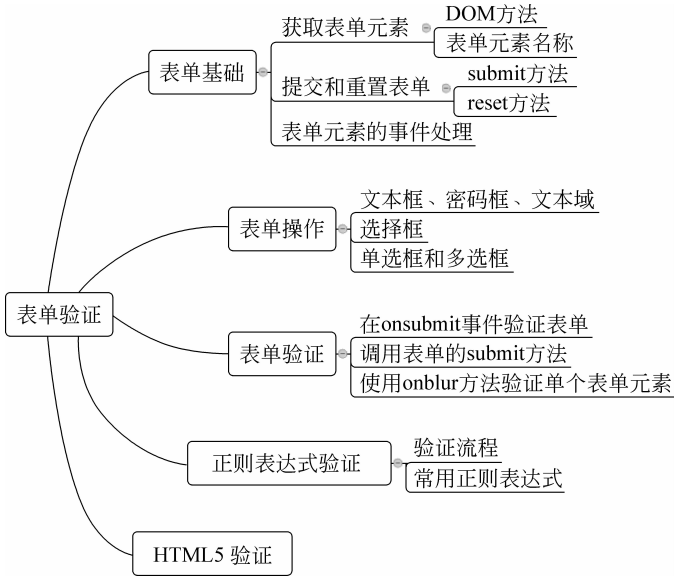
【本章要点】

- 表单元素
- 表单和表单元素的操作
- 表单数据验证的方法

【学习目标】

通过本章的学习掌握如何获取和操作表单和表单元素，掌握如何通过相应的事件来验证表单数据的有效性，会使用正则表达式来验证数据。

【思维导图】



JavaScript 可用来在数据被送往服务器前对 HTML 表单中的这些输入数据进行验证。在页面表单中要经常做如下的验证：

- 用户是否已填写表单中的必填项目？
- 用户输入的邮件地址是否合法？
- 用户是否已输入合法的日期？
- 用户是否在数据域（numeric field）中输入了文本？

5.1 表单基础

5.1.1 HTML表单

表单元素都是放在＜form＞＜/form＞标签内的。表单元素有三个重要的属性：

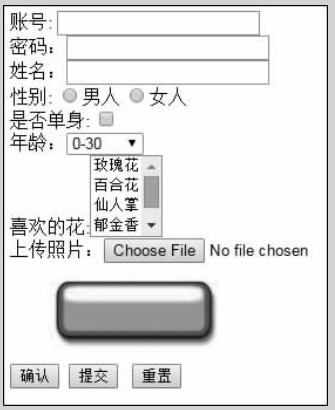
(1) action 指定该表单发送时接受操作的地址。

(2) method 指定表单数据发送的方法。可选值为 get、post。get 发送则表单内的数据将附加到 url 后发送。post 则是在 HTTP 请求中发送。

(3) enctype 指定表单数据在发送到服务器之前如何编码，特别要注意的是，当含有文件上传域时要设置编码方式为 enctype＝"multipart/form-data"，否则后台无法获取到浏览器发送的文件数据。默认情况下，enctype 值是 application/x-www-form-urlencoded，不能用于文件上传；只有使用了 multipart/form-data，form 里面的文件数据才以二进制的方式传递给服务器。

代码清单：5-1

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>表单基础</title>
</head>
<body>
<form id="form1" action="post.jsp" method=
"post" enctype="multipat/forn-data">
  <input type="hidden" value="隐藏信息"/>
   账号:<input type="text" maxlength="8"/><br/>
   密码:<input type="password"/><br/>
   姓名:<input type="text" name="name"/><br/>
  性别:<input type="radio" name="male"/>男人
       <input type="radio" name="male"/>女人<br/>
  是否单身:<input type="checkbox" name="single"/><br/>
  年龄:<select name="age">
         <option value="0">0-30</option>
         <option value="1">31-60</option>
       <option value="2">60-100</option>
     </select><br/>
  喜欢的花:<select multiple="multiple" name="flower">
     <option value="0">玫瑰花</option>
     <option value="1">百合花</option>
     <option value="2">仙人掌</option>
```



```
        <option value="3">郁金香</option>
        <option value="4">万寿菊</option>
  </select><br/>
  上传照片:<input type="file" /><br/>
  <input type="image" src="images/btn.jpg" /><br/>
  <input type="button" value="确认" />   <input type="submit"
  value="提交" />
    <input type="reset" value="重置" /><br/>
</form>
</body>
</html>
```

5.1.2　获取表单和表单元素

5.1.2.1　DOM 方法

可以像访问页面中的其他元素一样,使用原生 DOM 方法访问表单元素。

```
//查找 id 为 form1 的表单元素
var form1=document.getElementById("form1");
//查找 id 为 name 的 input 标签
var inputName=document.getElementById("name");
```

每个表单元素都有 elements 属性,该属性是表单中所有元素的集合。这个 elements 集合是一个有序列表,其中包含着表单中的所有字段,例如<input>、<textarea>、<button>和<fieldset>。每个表单字段在 elements 集合中的顺序与它们出现在表单中的顺序相同,可以按照位置和 name 特性来访问它们。

```
var form=document.getElementById("form1");
//取得表单中的第一个字段
var field1=form.elements[0];
//取得名为 textbox1 的字段
var field2=form.elements["textbox1"];
//取得表单中包含的字段的数量
var  filedCount=form.elements.length;
```

如果有多个表单控件的 name 属性值都一样(如一组单选按钮 name 属性值一般都是相同的),那么就会返回该 name 命名的一个 NodeList。例如,以下面的 HTML 代码片段为例。

```
<form method="post" id="myForm">
    <url>
        <li><input type="radio" name="color" value="red">Red</li>
```

```
        <li><input type="radio" name="color" value="green">Green</li>
        <li><input type="radio" name="color" value="blue">Blue</li>
    </url>
</form>
```

在这个 HTML 表单中，有 3 个单选按钮，它们的 name 都是 color，意味着在访问 elements["color"]时将会返回一个数组，其中包含这 3 个元素：

```
var form=document.GetElementById("myform");
var colorfields=form.elements["color"];
alert(colorfields.length);     //3
var firstcolorfield=colorfields[0];
var firstformfield=form.elements[0];
alert(firstcolorfield===firstformfield); //true
```

以上代码显示，通过 form. elements[0]访问的第一个表单字段与包含在 form. elements["color"]中的第一个元素相同。

5.1.2.2 通过表单元素名称访问

每个表单字段，不论它是按钮、文本框还是其他内容，均应包含在表单中，一般情况下需要给表单元素 name 赋予值，可以通过名字直接访问该表单元素。

```
document.form1.username;  //form1 是表单 form 的名称,username 是表单元素的名称
```

5.1.3 共有的表单字段属性和方法

除了＜fieldset＞元素之外，所有表单字段都拥有相同的一组属性。由于＜input＞类型可以表示几种表单字段，因此有些属性只适用于某些字段，但还有一些属性是所有字段所共有的。表单字段共有的属性和方法如下。

- disabled：布尔值，表示当前字段是否被禁用。
- form：指向当前字段所属表单的指针，只读。
- name：当前字段的名称。
- readonly：布尔值，表示当前字段是否只读。
- tabindex：表示当前字段的切换(tab)序号。
- type：当前字段的类型，如 checkbox、radio，等等。
- value：当前字段将被提交给服务器的值。对文件字段来说，这个属性是只读的，包含着文件在计算机中的路径。

代码清单 5-2 演示了如何操作表单元素的属性。

代码清单：5-2

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>表单基础</title>
<script>
function printForm(){
   var  elements = document.
   form1.elements;
  var info="";
  for (var i = 0; i < elements.
  length;i++){
    info=info+"elements
    ["+i+"]》";
    info=info+"name=
    "+elements[i].name+",";
    info=info+"type="
    +elements[i].type+",";
      info=info+"value="+elements[i].value+",";
      info=info+"disabled="+elements[i].disabled+",";
      info=info+"form="+elements[i].form+"<br>";
  }
  document.getElementById("p").innerHTML=info;
}
</script>
</head>
<body onload="printForm()">
<p id="p"></p>
<form id="form1" name="form1" action="post.jsp" method="post"enctype=
"multipart/form-data">
  <input name="id" type="hidden" value="隐藏信息"/>
   账号:<input name="account" type="text" maxlength="8"/><br/>
   密码:<input name="password" type="password"/><br/>
   姓名:<input name="name" type="text" name="name"/><br/>
  性别:<input type="radio" name="gender" value="male"/>男人
       <input type="radio" name="gender" value="female"/>女人<br/>
  是否单身:<input name="married" type="checkbox" name="single"/><br/>
  年龄:<select name="age">
         <option value="0">0-30</option>
         <option value="1">31-60</option>
         <option value="2">60-100</option>
     </select><br/>
```

```
  喜欢的花:<select multiple="multiple" name="flower">
      <option value="0">玫瑰花</option>
      <option value="1">百合花</option>
      <option value="2">仙人掌</option>
      <option value="3">郁金香</option>
      <option value="4">万寿菊</option>
  </select><br/>
  上传照片:<input type="file" name="file"/><br/>
  <input type="image" src="images/btn.jpg" /><br/>
  <input type="button" value="确认" />     < input type = "submit"
value="提交" />
    <input type="reset" value="重置" /><br/>
</form>
</body>
</html>
```

每个表单字段都有两个方法：focus()和 blur()。其中，focus()方法用于将浏览器的焦点设置到表单字段，即激活表单字段，使其可以响应键盘事件。例如，接收到焦点的文本框会显示插入符号，随时可以接收输入。使用 focus()方法，可以将用户的注意力吸引到页面中的某个部位。例如，在页面加载完毕后，将焦点转移到表单中的第一个字段。为此，可以监听页面的 load 事件，在该事件发生时，通过调用表单第一个字段的 focus()方法将焦点移到该元素上。例如：

```
window.onload=function (){
    document.userform.username.focus();
}
```

要注意的是，如果第一个表单字段是一个＜input＞元素，且其 type 特性的值为 hidden，那么以上代码会导致错误。另外，如果使用 CSS 的 display 和 visibility 属性隐藏了该字段，同样也会导致错误。

5.1.4 提交和重置表单

5.1.4.1 提交表单

用户单击提交按钮或图像按钮时，浏览器就会将该表单提交给服务器。使用＜input＞或＜button＞都可以定义提交按钮，只要将其 type 特性的值设置为 submit 即可，而图像按钮则是通过将＜input＞的 type 特性值设置为 image 来定义的。因此，只要单击以下代码生成的按钮，就可以提交表单。

```
<---通用提交按钮---->
<input type="submit" value="submit form">
<----自定义提交按钮---->
```

```
<button type="submit">submit form</button>
<----图像按钮---->
<input type="image" src="graphic.gif">
```

只要表单中存在上面列出的任何一种按钮，那么在任意一个表单控件拥有焦点的情况下，按回车键就可以提交该表单(textarea 是一个例外，在文本区中回车就会换行)。如果表单里没有提交按钮，那么按回车键不会提交表单。

以这种方式提交表单时，浏览器会在将请求发送给服务器之前触发 submit 事件。利用 submit 事件可以验证表单数据，并根据验证结果确定是否提交表单。通过阻止 onsubmit 事件的默认行为就可以取消表单的提交。例如：

```
var form=document. getElementById(" myform");
form.onsubmit=function(event){
    event=event || window.event;
    if(event.preventDefault) {
        event.preventDefault();
     } else {
         event.returnValue=false;
     }
}
```

第二种提交表单的方式通过调用表单对象的 submit 方法来提交数据。可以在编写代码的时候根据验证的结果来确定是否调用该方法。

```
var form=document.getElementById(myform");
form.submit();
```

5.1.4.2　重置表单

在用户单击重置按钮时，表单会被重置。使用 type 特性值为 reset 的＜input＞或＜button＞都可以创建重置按钮，如下面的例子所示。

```
<!----通用重置按钮---->
<input type="reset" value="reset form">
<!-自定义重置按钮 -->
<button type="reset">reset form</button>
```

这两个按钮都可以用来重置表单。在重置表单时，所有表单字段等会恢复到页面刚加载完毕时的初始值。如果某个字段的初始值为空，就会恢复为空；而带有默认值的字段，也会恢复为默认值。

用户单击重置按钮重置表单时，会触发 reset 事件。利用这个事件，可以在必要时取消重置操作。例如，下面展示了阻止重置表单的代码。

```
var form=document. getElementById(" myform");
form.onreset=function(event){
    event=event || window.event;
    if(event.preventDefault) {
        event.preventDefault();
     } else {
         event.returnValue=false;
     }
}
```

与提交表单一样，也可以通过 JavaScript 来重置表单，如下面的例子所示：

```
var form=document. getElementById(" myform");
form.reset();
```

5.1.5 表单元素的事件处理

除了支持鼠标、键盘、更改 HTML 事件之外，所有表单字段都支持下列 3 个事件。

- blur：当前字段失去焦点时触发。
- change：对于＜input＞和＜textarea＞元素。在它们失去焦点且 value 值改变时触发；对于＜select＞元素，在其选项改变时触发。
- focus：当前字段获得焦点时触发。

当用户改变了当前字段的焦点，或者调用了 blur()或 focus()方法时，都可以触发 blur 和 focus 事件。这两个事件在所有表单字段中都是相同的。但是，change 事件在不同表单控件中触发次数会有所不同。对于＜input＞和＜textarea＞元素，当它们才获得焦点到失去焦点且 value 值改变时才会触发 change 事件。对于＜select＞元素，只要用户选择了不同的选项，就会触发 change 事件。换句话说，不失去焦点也会触发 change 事件。

通常，可以使用 focus 和 blur 事件来以某种方式改变用户界面，要么是向用户给出视觉提示，要么是向界面中添加额外的功能（例如，为文本框显示一个下拉选项菜单）。而 change 事件则经常用来验证用户在字段中输入的数据。例如，假设有一个文本框，我们只允许用户输入数值。此时，可以利用 focus 事件修改文本框的背景颜色，以便更清楚地表明这个字段获得了焦点。可以利用 blur 事件恢复文本框的背景颜色，利用 change 事件在用户输入了非数值字符时再次修改背景颜色。下面就给出了实现上述功能的代码。

在下面的例子 onfocus 事件处理程序将文本框的背景颜色修改为黄色，以清楚地表示当前字段已经触发。随后，onblur 事件处理程序则会在发现非数值字符时，将文本框背景颜色修改为红色，为了测试用户输入的是不是非数值，这里针对文本框的 value 属性使用了简单的正则表达式。为确保无论文本框的值如何变化，验证规则始终如一，onblur 和 onchange 事件处理程序中使用了正则表达式。

代码清单：5-3


```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>表单元素事件处理</title>
<script>
var digitReg=new RegExp("^\\d+$");
function focusAge(edAge){
  edAge.style.borderColor="yellow";
}
function blurAge(edAge){

if (edAge.value.match(digitReg)==null){
    edAge.style.backgroundColor='red';
    edAge.focus();      //让焦点继续留在年龄的编辑框中
}else{
    edAge.style.backgroundColor='';
    edAge.style.borderColor="";
}
}
</script>
</head>
<body>
<p id="p"></p>
<form id="form1" name="form1" action="post.jsp" method="post">
  <p>年龄:<input name="age" type="text" maxlength="8" onblur="blurAge
(this)" onfocus="focusAge(this)" onchange="changeAge(this)"/>
  <p>名称:<input name="name" type="text" maxlength="8" />
</form>
</body>
</html>
```

5.2　表单操作

5.2.1　文本框、多行文本框和密码框

文本框、多行文本框和密码框这三个控件非常相似，而且多数时候的行为也差不多。

1. 选择内容

上述两种文本框都支持 select()方法，这个方法用于选择文本框中的所有文本。调

用 select()方法时,大多数浏览器(Opera 除外)都会将焦点设置到文本框中。这个方法不接受参数,可以在任何时候被调用。下面来看一个例子。

```
var textbox=document.form.textbox1;
textbox.select();
```

在鼠标单击文本框的时候,选择其所有的文本,这是一种非常常见的做法,特别是在文本框包含默认值的时候被调用。因为这样做就可以让用户不必一个一个地删除文本。下面展示了实现这一操作的代码。

代码清单:5-4


```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>select 全选</title>
<script>
function selectAll(textbox){
  textbox.focus();
  textbox.select();
}
</script>
</head>
<body>
<p id="p"></p>
<form id="form1" name="form1" action="post.jsp" method="post">
  <p>电话号码:<input name="telephone" type="text" value="请输入电话号码"
   onmouseup="selectAll(this)" />
</form>
</body>
</html>
```

2. 屏蔽字符

有时候,我们需要用户输入的文本中包含某些字符。例如,电话号码中不能包含非数值字符。如前所述,响应向文本框中插入字符操作的是 keypress 事件。因此,可以通过阻止这个事件的默认行为来屏蔽此类字符。如果只想屏蔽特定的字符,则需要检测 keypress 事件对应的字符编码,然后再决定如何处理。例如,下列代码只允许用户输入数字。

代码清单:5-5

```
<!DOCTYPE HTML>
<html>
<head>
```