**3. 表单验证**

表单验证是一套系统，它为终端用户检测无效的数据并标记这些错误，是一种用户体验的优化，让web应用更快的抛出错误，但它仍不能取代服务器端的验证，重要数据还要要依赖于服务器端的验证，因为前端验证是可以绕过的。

目前任何表单元素都有八种可能的验证约束条件：

|  |  |  |
| --- | --- | --- |
| 名称 | 用途 | 用法 |
| valueMissing | 确保控件中的值已填写 | 将required属性设为true，  <input type**=**"text"required**=**"required"/> |
| typeMismatch | 确保控件值与预期类型相匹配 | <input type**=**"email"/> |
| patternMismatch | 根据pattern的正则表达式判断输入是否为合法格式 | <input type**=**"text" pattern**=**"[0-9]{12}"/> |
| toolong | 避免输入过多字符 | 设置maxLength，<textarea id**=**"notes" name**=**"notes" maxLength**=**"100"></textarea> |
| rangeUnderflow | 限制数值控件的最小值 | 设置min，<input type**=**"number" min**=**"0" value**=**"20"/> |
| rangeOverflow | 限制数值控件的最大值 | 设置max，<input type**=**"number" max**=**"100" value**=**"20"/> |
| stepMismatch | 确保输入值符合min,max,step的设置 | 设置max min step，<input type**=**"number" min**=**"0" max**=**"100" step**=**"10" value**=**"20"/> |
| customError | 处理应用代码明确设置能计算产生错误 | 例如验证两次输入的密码是否一致，等会DEMO细说 |

下面展现浏览器自带的验证功能请在Chrome、Opera或Firefox中查看：

源代码：

|  |
| --- |
| <form name="register1" id="register1">            <p><label for="runnername">RunnerName:</label>               <input id="runnername"name="runnername" type="text" placeholder="First and last name"  required="required" autofocus="autofocus"/>            </p>            <p><label for="phone">Tel #:</label>               <input id="phone" name="phone" type="text" pattern="\d{3}-\d{4}-\d{4}"                      placeholder="xxx-xxxx-xxxx"/></p>            <p><label for="emailaddress">E-mail:</label>               <input id="emailaddress" name="emailaddress" type="email"                      placeholder="For confirmation only"/></p>            <p><label for="dob">DOB:</label>               <input id="dob" name="dob" type="date"                      placeholder="MM/DD/YYYY"/></p>            <p>Count:<input type="number" id="count" name="count" min="0" max="100" step="10"/></p>            <p><label for="style">Shirt style:</label>                 <input id="style" name="style" type="text" list="stylelist" title="Years of  participation"                  autocomplete="off"/></p>              <datalist id="stylelist">               <option value="White" label="1st Year"/>               <option value="Gray" label="2nd - 4th Year"/>               <option value="Navy" label="Veteran (5+ Years)"/>              </datalist>             <fieldset>              <legend>Expectations:</legend>              <p>              <label for="confidence">Confidence:</label>              <input id="confidence" name="level" type="range"                     onchange="setConfidence(this.value)"                     min="0" max="100" step="5" value="0"/>              <span id="confidenceDisplay">0%</span></p>              <p><label for="notes">Notes:</label>                 <textarea id="notes" name="notes" maxLength="100"></textarea></p>           </fieldset>             <p><input type="submit" name="register" value="Submit" onclick=" checkForm()"/></p>          </form> |

可是各个浏览器验证行为不一致，我们可能需要统一其验证行为，借助javascript我们可以统一浏览器的验证行为。

还是以上上述HTML为基础，我们为其加上相关javascript:

|  |
| --- |
| //自定义表单控件验证行为              var checkvalue = function(e){                  var el = e.target;                  var isvalid = el.checkValidity();                  if(isvalid){                      el.className= "";                      el.parentElement.getElementsByTagName("label")[0].className="";                  }else{                      el.className= "error";                      el.parentElement.getElementsByTagName("label")[0].className="error";                  }                  e.stopPropagation();                  e.preventDefault();              }              //定义表单验证方法              function invalidHandler(evt) {                  checkvalue(evt);              }              function loadDemo() {                 var myform = document.getElementById("register1");                 //注册表单的oninvlid事件                 myform.addEventListener("invalid", invalidHandler, true);                  for(var i=0;i< myform.elements.length-1;i++){                      //注册表单元素的onchange事件，优化用户体验                      myform.elements[i].addEventListener("change",checkvalue,false);                  }              }              //在页面初始化事件（onload）时注册的自定义事件              window.addEventListener("load", loadDemo, false); |

最后说下输入两次密码匹配的验证，写的很简单：

|  |
| --- |
| <form name="passwordChange">      <p><label for="password1">New Password:</label>      <input type="password" id="password1" onchange="checkPasswords()"></p>      <p><label for="password2">Confirm Password:</label>      <input type="password" id="password2" onchange="checkPasswords()"></p>  </form>  <button onclick="document.passwordChange.password1.checkValidity()">Check Validity</button> |
| function checkPasswords() {          var pass1 = document.getElementById("password1");          var pass2 = document.getElementById("password2");            if (pass1.value != pass2.value)              pass1.setCustomValidity("两次输入的密码不匹配");          else              pass1.setCustomValidity("");      } |