JavaScript

表单操作与正则表达式

目录

1 表单操作元素 3

1.1 文本域事件 3

1.2 按钮事件 3

1.3 复选框事件 3

1.4 单选按钮事件 5

1.5 列表框事件 6

2. 正则表达式 7

2.1 什么是正则表达式 7

2.2 正则表达式作用 7

2.3 正则语法 9

2.4 正则表达式对象的语法 10

# 表单操作元素

## 文本域事件

事件：onFocus onBlur onChange onInput

用法：

|  |
| --- |
| <input type="text" name="名称" size="宽度"  onFocus="接受焦点处理例程; "  onBlur="失去焦点处理例程; "  onChange="改变内容处理例程; “  onInput="输入内容处理例程"> |

## 按钮事件

事件：onClick

用法：

|  |
| --- |
| <input type="button" value="值" name="名称"  onClick="单击事件处理例程; "  > |

## 1.3 复选框事件

事件：onClick

用法：

|  |
| --- |
| <input type="button" value="值" name="名称"  onClick="单击事件处理例程; "  > |

示例：

HTML

|  |
| --- |
| <form name="myfm" >  <input type="checkbox" id="tt" name="checkbox" value="Java" />Java <br />  <input type="checkbox" name="checkbox" value="JavaScript" />JavaScript <br />  <input type="checkbox" name="checkbox" value="Ajax" />Ajax <br />  <input type="button" value="what's your choise?" onclick="choise()" />  <input type=“button” value=“把第一个变成不可选" onclick="disable()" />  </form> |

JavaScript

|  |
| --- |
| var chose = document.querySelectorAll("input[type=checkbox]");  console.log(chose);  function choise() {  var str = "";  for(var i=0;i<=chose.length-1;i++) {  if (chose[i].checked == true) {  str =str+ chose[i].value+"\n"; }  }  alert("You chose: "+"\n"+str);  }  function disable() {  chose[0].disabled = true;  } |

## 单选按钮事件

事件：onClick

用法：

|  |
| --- |
| <input type="radio" value="值" name="名称" checked  onClick="单击事件处理例程; "  > |

示例

HTML

|  |
| --- |
| <form name="myfm">  <input type="radio" id="tt" name="radio" value="Java"/>Java<br/>  <input type="radio" name="radio" value="JavaScript"/>JavaScript<br/>  <input type="radio" name="radio" value="Ajax"/>Ajax<br/>  <input type="button" value="what's your choise?" onclick="choise()"/>  <input type="button" value="把第一个变成不可选" onclick="disable()"/>  </form> |

JavaScript

|  |
| --- |
| var chose = document.querySelectorAll("input[type=radio]");  function choise() {  var chose = document.getElementsByName("radio");  var str = "";  for (var i = 0; i <= chose.length - 1; i++) {  if (chose[i].checked == true)  str = chose[i].value;  }  alert("You chose: " + str);  }  function disable() {  chose[0].disabled = true;  } |

## 列表框事件

事件：onFocus onBlur onChange

用法：

|  |
| --- |
| <select size="" name="" onFocus="" onBlur="" onChange="">  <option value="值1">值1</option>  <option value="值2">值2</option>  <option value="值3" selected>值3</option>  ……………………………  </select> |

示例

|  |
| --- |
| <form name="the\_form">  <select name="choose\_category"  onchange="show(the\_form.choose\_category.options  [selectedIndex].text);">  <option selected="selected">狗</option>  <option>鱼</option>  <option>鸟</option>  </select>  </form> |

JavaScript

|  |
| --- |
| <script type="text/javascript">  function show(txt) {  alert("选中的是" + txt);  }  </script> |

# 2. 正则表达式

## 2.1 什么是正则表达式

正则表达式就是用来描述被称为“正则集代数”的表达式，因此采用“正则表达式”这个术语。

## 2.2 正则表达式作用

测试字符串的某个模式，即数据有效性验证

实现按照某种规则替换文本

根据模式匹配从字符串中提取一个子字符串

2.3 正则表达式应用



## 2.3 正则语法

|  |  |
| --- | --- |
| 字符 | 描述 |
| \ | 转义 |
| ^ | 匹配输入字符串的开始位置 |
| $ | 匹配输入字符串的结束位置 |
| \* | 匹配前面的子表达式零次或多次。例如，zo\* |
| + | 匹配前面的子表达式一次或多次。例如，zo+ |
| ? | 匹配前面的子表达式零次或一次。例如，do? |
| {*n*} | *n* 是一个非负整数。匹配确定的 *n* 次。例如，o{2} |
| {*n*,} | *n* 是一个非负整数。至少匹配*n* 次。例如，o{2,} |
| {*n*,*m*} | *m* 和*n* 均为非负整数，其中*n*<=*m* 。最少匹配 *n* 次且最多匹配 *m* 次。例如：o{1,3} |
| *x*|*y* | 匹配 *x* 或 *y*。例如，z|food |
| [*xyz*] | 字符集合。匹配所包含的任意一个字符。例如：[abc] |
| [^*xyz*] | 负值字符集合。匹配未包含的任意字符。例如：[^abc] |
| [*a-z*] | 字符范围。匹配指定范围内的任意字符 |
| [^*a-z*] | 负值字符范围。匹配任何不在指定范围内的任意字符 |
| \b | 匹配一个单词边界。例如: er\b 可以匹配"never" |
| \B | 匹配非单词边界。 |
| \d | 匹配一个数字字符。等价于 [0-9]。 |
| \D | 匹配一个非数字字符。等价于 [^0-9] |
| \n | 匹配一个换行符 |
| \r | 匹配一个回车符 |
| \s | 匹配任何空白字符 |
| \S | 匹配任何非空白字符 |
| . | 匹配除 "\n" 之外的任何单个字符 |

## 2.4 正则表达式对象的语法

|  |  |
| --- | --- |
| **方法** | **含义** |
| **exec** | **在字符串中运行查找，并返回包含该查找结果的一个数组** |
| **test** | **返回一个 Boolean 值，被查找的字符串中是否存在某模式** |
| **search** | **返回与正则表达式查找内容匹配的第一个子字符串的位置** |
| **split** | **按特定字符串将原字符串拆分成数组** |
| **match** | **在字符串中执行查找，并将包含查找的结果作为数组返回** |

示例

用户名检测

|  |
| --- |
| <script type="text/javascript">  function check(val) {  var regx = /[a-z0-9]\w\d$/g  //不包含规定字符就不合法  if (regx.test(val)) {  alert(“\n用户名合法\n");  }  else  alert(“\n用户名不合法 \n");  }  </script>  <form name="the\_form">  用户名:<input type="text" name="text1" onblur="check(this.value)" />  </form> |

手机号检测

|  |
| --- |
| <script type="text/javascript">  function check(val) {  var regx = /^(\+86)?13\d{9}$/;  var obj = document.getElementById("lable");  if (regx.test(val)) {  obj.style.color = “green“; obj.innerHTML = ”√“;  }  else {  obj.style.color = “red“; obj.innerHTML = ”号码不合法，请重新输入“;  }  }  </script>  <form name="the\_form">  手机号码：<input type="text" name="text1" onblur="check(this.value)" />  <a name="lable" id="lable" ></a>  </form> |

2.5 常用正则表达式

数字：^[0-9]\*$

中国邮政编码：[1-9]\d{5}(?!\d)

身份证号：^\d{15}|\d{18}$

Email地址：^\w+([-+.]\w+)\*@\w +([-.]\w+)\*\.\w+([-.]\w+)\*$

示例exce

|  |
| --- |
| function isDigit(s){  var reg = /^[0-9]{1,20}$/;  var result = reg.exec(s);  //如果格式正确，返回原字符串，否则返回null  alert(result);  } |

示例search

|  |
| --- |
| function seachString(){  var r, re; // 声明变量  var s = "The rain in Spain falls mainly in the plain.";  re = /falls/; // 创建正则表达式模式  r = s.search(re); // 查找字符串  alert(r); // 返回 int 结果表示出现位置  } |

示例split

|  |
| --- |
| function splitString( ){  var r, reg; // 声明变量  reg = /abc/; // 创建正则表达式模式  var s = “IcansaymyabcIcansaymyabcIcan…";  r = s.split(reg); // 分割字符串  for(var p in r){ // 返回 String 数组  alert(r[p]);  }  } |

示例match

|  |
| --- |
| function matchDemo(String source){  var reg = /^[a-z]/;  var result = source.match(reg);  alert(result);  } |