返回目录

第九章　正则表达式

本章导读

正则表达式(英语：Regular Expression、简写为regex或regexp)在字符串描述以及搜索匹配方面有非常重要的应用。正则表达式使用单个字符串以描述或匹配一系列符合某个句法规则的字符串。在很多文本编辑器，正则表达式通常被用来检索、替换符合正则规则的文本。许多程序设计语言也都支持利用正则表达式进行字符串操作。

学习目标：

1. 了解正则表达式及其应用场景；
2. 掌握常用的正则表达式；

本章目录

第一节　快速入门

在String(字符串)对象中的match()、replace()、search()、split()均支持正则表达式，在正则表达式的支持下，这些方法(函数)功力将大为提升，如例程9-1所示。

例程9-1

```
第1行   <Html>
第2行       <Head>
第3行           <Title>正则表达式</Title>
第4行           <Script language="JavaScript" src="jQuery-1.9.1.min.js"></Script>
第5行       </Head>
第6行       <Body>
第7行           <Script language="JavaScript">
第8行               var content="计算机可分为超级计算机、工业控制计算机、网络计算机……";
第9行               var newContent1=content.replace("计算机","电脑");
第10行              alert(newContent1);//仅第一个计算机被替换为电脑
第11行
第12行              var newContent2=content.replace(/计算机/ig,"电脑");
第13行              alert(newContent2);//所有计算机全部被替换为电脑
第14行          </Script>
```

```
第15行          </Body>
第16行      </Html>
```

比较例程9-1中第9行与第12行，会发现replace()的第一个参数有些不同，第9行的“计算机”三字在双引号内，是为字符串对象(String)；第12行的“计算机”三字在反斜杠之内，是为正则表达式对象(RegExp)，斜杠后的ig是正则表达式对象的参数，i表示忽略大小写，在处理英文字符时有用，g则表示全局适用，即对整个字符串适用，不是匹配一次即终止。例程9-1的功能如果不采用正则表达式，则需要采用循环方式，逐个将“计算机”替换为电脑。由此也可以看出，正则表达式的基本应用较为简单，但功力却十分强大。

正则表达式和Math、String、Array等一样，都是JavaScript的基本对象，其对象名称为RegExp。如同Array对象可以用方括号如[1,4,6,10,2]简捷表示数组对象，双反斜杠和其间内容也构成了正则表达式对象，如例程9-1第12行所示，正则表达式对象还可以用例程9-2所示方式申明。

例程9-2

```
第1行    <Script language="JavaScript">
第2行        var content="计算机可分为超级计算机、工业控制计算机、网络计算机……";
第3行
第4行        var myRegExp1=new RegExp("计算机","ig");//申明正则表达式对象
第5行        var myRegExp2=/计算机/ig;//正则表达式的简捷申明方式，功能与上行一样
第6行
第7行        var newContent1=content.replace(myRegExp1,"电脑");
第8行        alert(newContent1);//所有计算机全部被替换为电脑
第9行
第10行       var newContent2=content.replace(myRegExp2,"电脑");
第11行       alert(newContent2);//所有计算机全部被替换为电脑
第12行   </Script>
```

例程9-3是正则表达式用于“邮政编码”判断的例子。在网页中，常常要求输入邮政编码，为了符合规则，需要对内容进行校验。在第4行，斜杠后的^表示开始，斜杠前的$表示结束，即开始为[1-9],结束为[0-9]。方括号[]表示字符范围，[1-9]表示字符为从1开始到9结束，[0-9]表示从0开始到结束。{4}表示重复次数，此处为4次重复，重复内容由{4}前的紧邻内容决定，此处[0-9]{4}表示0-9范围内的数字重复4次。从上述正则表达式可以看出，邮政编码的规则必须以1-9开始，中间为0-9之间的数字，结尾前也为0-9，开始和结束都必须为数字，其长度为6(开1-9，占1个字符，中间4个0-9占4个字符，结束0-9占一个字符，总计6个字符)。

例程9-3

```
第1行    <Script language="JavaScript">
第2行        var postCode="100871";
第3行
第4行        var myRegExp=/^[1-9][0-9]{4}[0-9]$/ig;//邮政编码正则表达式
第5行
第6行        var execResult=myRegExp.test(postCode);//正则表达式对象的方法test()
第7行
第8行        alert(execResult);//显示true，test()方法执行结果保存在execResult中
第9行    </Script>
```

例程9-3第6行之test()为正则表达式对象的方法，其功能是判断test()方法的字符串参数是否有符合正则表达式的内容，如有则返回true，否则返回false。

例程9-3第4行的正则表达式可以修改为/^[1-9]\d{5}$/ig，其中\d代表数字，即0-9。类似\d的还有\w表示单词字符，如英文的a……z和A……Z。表示任意字符则用英文句点(.)。如将/^[1-9]\d{5}$/ig修改为/^[1-9].{5}$/ig则表示1-9开始到结束符之间5个任意字符(包括数字、英文字符等)。

第二节　字符范围限制

字符范围限制主要由方括号完成，如例程9-3第4行中的/^[1-9][0-9]{4}[0-9]$/ig方括号就是用于限制字符范围。更多字符范围限制方式如下表所示。

表1：正则表达式对象字符范围限制

表达式	描述
[abc]	限制为方括号内指定的字符，abc可以换成其他英文字符和数字。
[^abc]	限制为不是方括号内指定的字符。
[0-9]	限制为0-9之间的数字。
[a-z]	限制为a-z之间的英文字符，起点字符和重点字符可以调整，但其间为连续。
[A-Z]	限制为A-Z之间的英文字符，起点字符和重点字符可以调整，但其间为连续。
[A-z]	限制为大写 A 到小写 z 的字符，起点字符和重点字符可以调整，但其间为连续。

第三节　元字符

元字符就似乎有特殊含义的字符，如/^[1-9]\d{5}$/ig中的\d就是元字符，代表所有数字字符。正则表达式中还有不少元字符，如下表所示。

元字符	描述
.	代表单个字符，除了换行和行结束符。
\w	代表单词字符，如英文字母等。
\W	代表非单词字符，如数字、$、#等。
\d	代表数字，如0-9。
\D	代表非数字字符，如各种字符、符号等
\s	代表空白字符，如空格等。
\S	代表非空白字符，如字符、数字、符号等。
\b	代表单词边界，不匹配任何字符。\b只是一个位置，一侧是构成单词的字符，另一侧为非单词字符、字符串的开始或结束位置。\b是零宽度的。
\B	代表非单词边界。
\0	代表 NUL字符。
\n	代表换行符。
\f	代表换页符。
\r	代表回车符。
\t	代表制表符。
\v	代表垂直制表符。
\xxx	代表以八进制数xxx规定的字符。
\xdd	代表以十六进制数dd规定的字符。
\uxxxx	代表以十六进制数xxxx规定的 Unicode 字符。

第四节　量词

量词用于限定字符出现的次数，如/^[1-9]\d{5}$/ig中的{5}就是就是量词，表示\d出现5次，更多量词如下表所示。

量词	描述
n+	表示n所代表的字符至少有一个。
n*	表示n所代表的字符有零个或多个。
n?	表示n所代表的字符有零个或一个。

n{X}	表示n所代表的字符有X个。
n{X,Y}	表示n所代表的字符有X或Y个。
n{X,}	表示n所代表的字符至少有X个。
n$	表示n所代表的字符其后为结尾。
^n	表示n所代表的字符在开始。
?=n	表示其后紧接着n所代表的字符。
?!n	表示其后没有紧接着n所代表的字符。

第五节　应用示例

1、网络帐号

网络帐号一般要求以字母开头，长度介于5-16个字符，仅允许字母数字和下划线，其正则表达式为：

```
^[a-zA-Z][a-zA-Z0-9_]{4,15}$
```

其含义是以[a-zA-Z]开始，在结束符$之前，出现[a-zA-Z0-9_]4到15次(总长5-16，减去1首字符，因此为4-15次)。

2、国内电话号码

国内电话号码有两部分构成，即区号和号码，其间用中横线联结，如0511-7998048或028-62751914，其正则表达式为：

```
\d{3}-\d{8}|\d{4}-\d{7}
```

其含义是以3个数字然后一个中横线后8位数字，或4个数字然后一个中横线后7为数字。

3、QQ号码

腾讯QQ号码从10000开始，长度至少为5位，其正则表达式为：

```
[1-9][0-9]{4,}
```

其含义是以1-9数字开始，然后紧跟至少4个数字。

4、整数

整数分为：正整数、负整数、整数(包含正负)、非负整数、非正整数，其正则表达式如下：

^[1-9]\d*$	匹配正整数，其含义是以1-9开始，然后跟多个数字。
^-[1-9]\d*$	匹配负整数，以负号开始，其后紧跟数字1-9，在其后是多个数字。
^-?[1-9]\d*$	匹配整数，-?表示可能有负号，也可以没有，其后跟数字1-9，在其后是多个数字。
^[1-9]\d*|0$	匹配非负整数，包括0
^-[1-9]\d*|0$	//匹配非正整数，包括0

第六节　正则表达式的方法

1、RegExp.exec()===index

RegExp.exec()方法用于检索字符串中正则表达式的匹配，如匹配成功则返回匹配内容，如没有找到返回值为null，如例程9-4。

例程9-4

```
<Script language="JavaScript">
        var content="计算机可分为超级计算机、工业控制计算机、网络计算机……";
        var myRegExp=/计算机/ig;

        var result=myRegExp.exec(content);//result保存匹配结果

        while(result!=null){//如果没有匹配成功，则result值为null
                document.write(result.lastIndex);//lastIndex匹配的起始位置，也可以说是本次查找的结束位置
                document.write(result);//输出匹配结果
                document.write("<br />");
                result=myRegExp.exec(content);//从lastIndex位置起继续匹配
```

```
第12行            }
第13行    </Script>
```

例程9-4工作原理是用正则表达式查找字符串对象content，一直循环查找，直到返回值为null为止。RegExp.exec()的返回值为对象，其属性lastIndex表明下一次匹配起始位置。

2、RegExp.test()

RegExp.test()方法用于检索字符串是否有与正则表达式匹配的内容，如果有则返回true，否则返回false，如例程9-5所示。

例程9-5

```
第1行    <Script language="JavaScript">
第2行            var content="While the wife was out, the husband tried to cook a meal in the kitchen, but he was just like a bull in a china shop";
第3行            var myRegExp=/china/ig;
第4行
第5行            var result=myRegExp.test(content);//result保存匹配结果,其结果为true
第6行            alert(result);//显示为true
第7行
第8行    </Script>
```

RegExp.test()也常用于判断输入是否正确，如例程9-6所示。当id名为postCode获得焦点时，其后提示信息删除，当离开焦点时(blur)则判断输入是否正确，如果正确其后显示打钩，如果错误，则打叉。

例程9-6

```
第1行    <Html>
第2行            <Head>
第3行                    <Title>正则表达式</Title>
第4行                    <Script language="JavaScript" src="jQuery-1.9.1.min.js"></Script>
第5行                    <Style type="text/css">
第6行                            .hint{font-size:12px;color:red}
第7行                    </Style>
第8行            </Head>
第9行            <Body>
第10行                   <Div>学生姓名<input type=text id="studName"></Div>
第11行                   <Div>邮政编码<input type=text id="postCode"></Div>
第12行                   <Script language="JavaScript">
第13行                           $("#postCode").blur(function(){
第14行                                   var strInput=$("#postCode").val();//#postCode代表邮政编码输入框
第15行
第16行                                   var myRegExp=/^[1-9]\d{5}$/ig;//以1-9数字开始紧接5个数字直到结束
第17行
第18行                                   var result=myRegExp.test(strInput);//result保存匹配结果
第19行
第20行                                   if (result==true){
第21行                                           $(this).parent().append("<span class='hint'>√</span>");
第22行                                   }else{
第23行                                           $(this).parent().append("<span class='hint'>×</span>");
第24行                                   }
第25行                           });
```

```
第26行
第27行                    $("#postCode").focus(function(){
第28行                        $(this).parent().children(".hint").remove();//获得焦点时删除提示信息
第29行                    });
第30行                </Script>
第31行            </Body>
第32行    </Html>
```

第七节　字符串对象应用正则表达式

在字符串对象(string)中，replace()、search()、match()以及split()支持正则表达式。在正则表达式的支持下，字符串对象的四种方法功力大为提升。

1、String.replace()方法

replace()在前面已经提及，其功能是找到字符串中匹配内容，并用指定的内容替换匹配内容，如例程9-7所示。

例程9-7

```
第1行    <Html>
第2行        <Head>
第3行            <Title>正则表达式</Title>
第4行            <Script language="JavaScript" src="jQuery-1.9.1.min.js"></Script>
第5行            <Style type="text/css">
第6行                .hint{font-size:12px;color:red}
第7行            </Style>
第8行        </Head>
第9行        <Body>
第10行            <Div>
第11行                唐朝（618—907年），是中国历史上统一时间最长，国力最强盛的朝代之一。
第12行                618年由李渊建立，定都长安（今西安）。627年，李世民登基后开创了“贞观之治”，
第13行                唐高宗以后，武则天一度迁都洛阳15年以周代唐（690-705年），史称武周，
第14行                705年唐中宗恢复大唐国号，还都长安。唐玄宗李隆基即位后，开创了全盛的“开元盛世”。
第15行                安史之乱后，国力日趋衰败。907年朱温篡唐，唐朝灭亡。唐朝共历289年，20位皇帝。
第16行                唐朝声誉远及海外，与南亚、西亚和欧洲国家均有往来。唐朝以后海外多称中国人为“唐人”。
第17行                唐诗、科技、文化艺术极其繁盛，具有多元化的特点。
第18行            </Div>
第19行            <Script language="JavaScript">
第20行                //本例删除文本中连接词、标点符号、数字等，在中文分词中或有使用
第21行
第22行                var content=$("Div").text();//获得div下的文本
第23行                var myRegExp=/\d|的|与|是|为|年|上|了|最|由|于|以|后|其|，|。|（|）|、|“|”|\-|—/ig;
第24行                content=content.replace(myRegExp," ");
第25行                $("Div").text(content);
第26行
第27行            </Script>
第28行        </Body>
第29行    </Html>
```

2、String.search()方法

search()方法用于探测符合正则表达式字符串第一次出现位置，该方法不执行全局匹配，哪怕是正则表达式中有g参数，并且lastIndex属性值也予以忽视，如例程9-8所示，其代码是替代例程9-7的JavaScript代码。

例程9-8

```
<Script language="JavaScript">
        var content=$("Div").text();//获得div下的文本
        var myRegExp=/唐代|唐朝|大唐/ig;
        var position=content.search(myRegExp);
        alert(position);//显示为
</Script>
```

3、String.match()方法

String.match()用于查找字符串中匹配正则表达式的情况，如果匹配成功，则返回匹配内容的数组的数组，如匹配失败，则返回null。是否执行全局匹配，与正则表达式是否设置g参数相关，如例程9-9所示，其代码是替代例程9-7的JavaScript代码。

例程9-9

```
<Script language="JavaScript">
        var content=$("Div").text();//获得div下的文本
        var myRegExp=/唐.{1}/ig;
        var myMatch=content.match(myRegExp);
        alert(myMatch);//唐朝,唐高,唐（,唐中,唐国,唐玄,唐，,唐朝,唐朝,唐朝,唐朝,唐人,唐诗
        alert(myMatch.length);//显示数组myMatch的长度，此处显示13
</Script>
```

4、String.split()方法

String.split()用于将字符串切分为数组，例程9-10是将字符串从标点符号处切分，是替代例程9-7的JavaScript代码。

例程9-10

```
<Script language="JavaScript">
        var content=$("Div").text();//获得div下的文本
        var myRegExp=/，|。|、/ig;
        var mySplit=content.split(myRegExp);//从逗号、句号和顿号出切分后数组长度
        alert(mySplit.length);//显示为29
</Script>
```

第八节　贪婪与懒惰匹配

先来看看例程9-11，其功能是匹配网页中的链接。

例程9-11

```
<Html>
    <Head>
        <Title>正则表达式</Title>
        <Script language="JavaScript" src="jQuery-1.9.1.min.js"></Script>
        <Style type="text/css">
            .hint{font-size:12px;color:red}
        </Style>
    </Head>
    <Body>
        <Div>
```

```
第11行                    搜索引擎是指根据一定的策略、运用特定的计算机程序从互联网上搜集信息，
第12行                    在对信息进行组织和处理后，为用户提供检索服务，将用户检索相关的信息
第13行                    展示给用户的系统。搜索引擎包括全文索引、目录索引、元搜索引擎、垂直搜索引擎、
第14行                    集合式搜索引擎、门户搜索引擎与免费链接列表等。国外知名搜索引擎有
第15行                    <A href="http://www.google.com">谷歌</A>，国内知名搜索引擎有：
第16行                    <A href="http://www.baidu.com">百度</A>、
第17行                    <A href="http://www.sogou.com">搜狗</A>、
第18行                    <A href="http://www.so.com">360搜索</A>等。
第19行                </Div>
第20行                <Script language="JavaScript">
第21行                    var content=$("Div").text();//获得div下的文本
第22行                    var myRegExp=/<a.*<\/a>/ig;//以<a开始，以</a>结束，中间有多个字符的正则表达式
第23行                    var myMatch=content.match(myRegExp);
第24行                    alert(myMatch.length);//显示为1
第25行                </Script>
第26行            </Body>
第27行    </Html>
```

阅读上述代码，会发现，有4个链接，但是执行上述JavaScript代码，却显示只有一个。当将正则表达式由/<a.*<\/a>/ig;改变为/<a.*?<\/a>/ig;则将显示为4，符合直观判断。原因是什么呢？这就涉及到正则表达式匹配时的贪婪模式和懒惰模式。

当出现重复数量(比如：*，？)的时候，会尽可能的多匹配，即为贪婪模式。上述的正则表达式中.表示任意字符，* 代表可以重复出现任意个，根据正则表达式默认的贪婪个性，一直匹配到最后一个为止。而增加?就可以让正则表达式变为懒惰模式，尽可能少且满足匹配条件即可。

贪婪模式，满足匹配条件且尽量多内容。贪婪模式可以理解为找到起点匹配内容后，从被匹配内容结束位置开始查找符合条件的匹配内容。懒惰模式，满足匹配条件但尽量少的内容。可以理解为在找到起点内容后，即从该位置开始查找符合条件的内容，符合找到符合条件内容后即停止或开始一次新的匹配。