**正则表达式**

拓薪教育

刘洋

1. 初识正则表达式
   1. 正则表达式概述

正则表达式，又称正规表示法、常规表示法（英语：Regular Expression，在代码中常简写为regex、regexp或RE），计算机科学的一个概念。正则表达式使用单个字符串来描述、匹配一系列符合某个句法规则的字符串。

正则表达式类似于JSON,是一种通用的标准,被各种开发语言所支持,包括但不限于:Java,JavaScript,C,C++,C#,Python,SQL等等;

因为在Javaweb项目中正则一般用于前端验证,故此我们使用JavaScript来进行正则表达式的学习;

* 1. 正则表达式使用

第一种方式: var reg = new RegExp('^a$')

|  |
| --- |
| ***var*** reg **= *new RegExp*('^a$');** // 第一种方式 |

第二种方式: var reg = /^a$/;

|  |
| --- |
| ***var*** reg **=** /^a$/**;** // 第二种方式 在js中推荐使用 |

* 1. 第一个例子

|  |
| --- |
| **<*html* lang="en"> <*head*>  <*meta* charset="UTF-8">  <*title*>**正则表达式**</*title*> </*head*> <*body*> <*input* type="text" id="name" />*&nbsp;&nbsp;* <*span* id="msg"></*span*><*br*> <*input* type="button" onclick="***test***()" value="测试"> </*body*> <*script* type="text/javascript">  *function*** *test***(){  *var*** value **= *document*.**getElementById**("name").*value*;** // 定义一个匹配字符a的正则表达式  // ***var*** reg **= *new RegExp*('^a$');** // 第一种方式  ***var*** reg **=** /^a$/**;** // 第二种方式 在js中推荐使用  // 获取正则匹配的结果  ***var*** result **=** reg**.**test**(**value**);** // 定义显示结果  ***var*** html **=** result **? '合法' : '不合法';** // 在页面显示正则匹配结果  ***document*.**getElementById**("msg").**innerHTML **=** html**;  } </*script*> </*html*>** |

1. 正则表达式的匹配规则
   1. 字符类匹配

* **[abc]** a、b 或 c 任意**一个**字符
* **[^abc]** 任何字符，除了 a、b 或 c ,单个字符
* **[a-zA-Z]** a 到 z 或 A 到 Z，两头的字母包括在内
* **[0-9]** 0-9之间的任意字符
  1. 元字符匹配
* **.** 匹配除换行符（\n、\r）之外的任何单个字符。要匹配包括 '\n' 在内的任何字符，请使用像"(.|\n)"的模式。
* **\w** 匹配字母、数字、下划线。等价于'[A-Za-z0-9\_]'。
* **\W** 匹配非字母、数字、下划线。等价于 '[^A-Za-z0-9\_]'。
* **\s** 匹配任何空白字符，包括空格、制表符、换页符等等。等价于 [ \f\n\r\t\v]。
* **\S** 匹配任何非空白字符。等价于 [^ \f\n\r\t\v]。
* **\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]。
* **\b** 匹配一个单词边界，也就是指单词和空格间的位置。例如， 'er\b' 可以匹配"never" 中的 'er'，但不能匹配 "verb" 中的 'er'。
* **^** 匹配输入字符串的开始位置。如果设置了 RegExp 对象的 Multiline 属性，^ 也匹配 '\n' 或 '\r' 之后的位置。
* **$** 匹配输入字符串的结束位置。如果设置了RegExp 对象的 Multiline 属性，$ 也匹配 '\n' 或 '\r' 之前的位置。
  1. 数量词
* X**?** 一次或一次也没有
* X**\*** 零次或多次
* X**+** 一次或多次
* X**{n}** 恰好 n 次
* X**{n,}** 至少 n 次
* X**{n,m}** 至少 n 次，但是不超过 m 次
  1. 字符转义

如果你想查找元字符本身的话，比如你查找.,或者\*,就出现了问题：你没法指定它们，因为它们会被解释成其它的意思。这时你就必须使用\来取消这些字符的特殊意义。因此，你应该使用\.和\\*。当然，要查找\本身，你也得用\\.

例如：www\.txjava\.cn匹配www.txjava.cn，c:\\windows匹配c:\windows.

* 1. 分组

我们已经提到了怎么重复单个字符（直接在字符后面加上限定符就行了）；但如果想要重复一个字符串又该怎么办？你可以用小括号来指定子表达式(也叫做分组)，然后你就可以指定这个子表达式的重复次数了，你也可以对子表达式进行其它一些操作(后面会有介绍)。

(\d{1,3}\.){3}\d{1,3}是一个简单的IP地址匹配表达式。要理解这个表达式，请按下列顺序分析它：\d{1,3}匹配1到3位的数字，(\d{1,3}\.){3}匹配三位数字加上一个英文句号(这个整体也就是这个分组)重复3次，最后再加上一个一到三位的数字(\d{1,3})。

不幸的是，它也将匹配256.300.888.999这种不可能存在的IP地址(IP地址中每个数字都不能大于255)。如果能使用算术比较的话，或许能简单地解决这个问题，但是正则表达式中并不提供关于数学的任何功能，所以只能使用冗长的分组，选择，字符类来描述一个正确的IP地址：

iPv4的ip地址都是（1~225）.（0~255）.（0~255）.（1~255）的格式

^(1\d{2}|2[0-4]\d|22[0-5]|[1-9]\d|[1-9])\.

(1\d{2}|2[0-4]\d|25[0-5]|[1-9]\d|\d)\.

(1\d{2}|2[0-4]\d|25[0-5]|[1-9]\d|\d)\.

(1\d{2}|2[0-4]\d|25[0-5]|[1-9]\d|[1-9])$

(1\d{2}|2[0-4]\d|22[0-5]|[1-9]\d|[1-9])\.(1\d{2}|2[0-4]\d|25[0-5]|[1-9]\d|\d)\.(1\d{2}|2[0-4]\d|25[0-5]|[1-9]\d|\d)\.(1\d{2}|2[0-4]\d|25[0-5]|[1-9]\d|[1-9])

1. 常见的正则表达式
   1. 数字校验
2. 数字：^[0-9]\*$
3. n位的数字：^\d{n}$
4. 至少n位的数字：^\d{n,}$
5. m-n位的数字：^\d{m,n}$
6. 零和非零开头的数字：^(0|[1-9][0-9]\*)$
7. 非零开头的最多带两位小数的数字：^([1-9][0-9]\*)+(.[0-9]{1,2})?$
8. 带1-2位小数的正数或负数：^(\-)?\d+(\.\d{1,2})?$
9. 正数、负数、和小数：^(\-|\+)?\d+(\.\d+)?$
10. 有两位小数的正实数：^[0-9]+(.[0-9]{2})?$
11. 有1~3位小数的正实数：^[0-9]+(.[0-9]{1,3})?$
12. 非零的正整数：^[1-9]\d\*$ 或 ^([1-9][0-9]\*){1,3}$ 或 ^\+?[1-9][0-9]\*$
13. 非零的负整数：^\-[1-9][0-9] \*$ 或 ^-[1-9]\d\*$
14. 非负整数：^\d+$ 或 ^[1-9]\d\*|0$
15. 非正整数：^-[1-9]\d\*|0$ 或 ^((-\d+)|(0+))$
16. 非负浮点数：^\d+(\.\d+)?$ 或 ^[1-9]\d\*\.\d\*|0\.\d\*[1-9]\d\*|0?\.0+|0$
17. 非正浮点数：^((-\d+(\.\d+)?)|(0+(\.0+)?))$ 或 ^(-([1-9]\d\*\.\d\*|0\.\d\*[1-9]\d\*))|0?\.0+|0$
18. 正浮点数：^[1-9]\d\*\.\d\*|0\.\d\*[1-9]\d\*$
19. 负浮点数：^-([1-9]\d\*\.\d\*|0\.\d\*[1-9]\d\*)$
20. 浮点数：^(-?\d+)(\.\d+)?$ 或 ^-?([1-9]\d\*\.\d\*|0\.\d\*[1-9]\d\*|0?\.0+|0)$
    1. 字符校验
21. 汉字：^[\u4e00-\u9fa5]{0,}$
22. 英文和数字：^[A-Za-z0-9]+$ 或 ^[A-Za-z0-9]{4,40}$
23. 长度为3-20的所有字符：^.{3,20}$
24. 由26个英文字母组成的字符串：^[A-Za-z]+$
25. 由26个大写英文字母组成的字符串：^[A-Z]+$
26. 由26个小写英文字母组成的字符串：^[a-z]+$
27. 由数字和26个英文字母组成的字符串：^[A-Za-z0-9]+$
28. 由数字、26个英文字母或者下划线组成的字符串：^\w+$ 或 ^\w{3,20}$
29. 中文、英文、数字包括下划线：^[\u4E00-\u9FA5A-Za-z0-9\_]+$
30. 中文、英文、数字但不包括下划线等符号：^[\u4E00-\u9FA5A-Za-z0-9]+$
31. 可以输入含有^%&',;=?$\"等字符：[^%&',;=?$\x22]+
32. 禁止输入含有~“”的字符：[^~\x22]+
    1. 特殊需求校验
33. Email地址：^\w+([-+.]\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*$
34. InternetURL：^http://([\w-]+\.)+[\w-]+(/[\w-./?%&=]\*)?$
35. 手机号码：^(13[0-9]|14[5|7]|15[0|1|2|3|5|6|7|8|9]|18[0|1|2|3|5|6|7|8|9])\d{8}$
36. 国内电话号码(0511-4405222、021-87888822)：\d{3}-\d{8}|\d{4}-\d{7}
37. 身份证号(15位、18位数字)：^\d{15}|\d{18}$
38. 短身份证号码(数字、字母x结尾)：^([0-9]){7,18}(x|X)?$ 或 ^\d{8,18}|[0-9x]{8,18}|[0-9X]{8,18}?$
39. 帐号是否合法(字母开头，允许5-16字节，允许字母数字下划线)：^[a-zA-Z][a-zA-Z0-9\_]{4,15}$
40. 密码(以字母开头，长度在6~18之间，只能包含字母、数字和下划线)：^[a-zA-Z]\w{5,17}$
41. 日期格式：^\d{4}-\d{1,2}-\d{1,2}
42. 一年的12个月(01～09和1～12)：^(0?[1-9]|1[0-2])$
43. 一个月的31天(01～09和1～31)：^((0?[1-9])|((1|2)[0-9])|30|31)$
44. 正则表达式的语法明细

|  |  |
| --- | --- |
| 字符 | 说明 |
| \ | 将下一字符标记为特殊字符、文本、反向引用或八进制转义符。例如，"n"匹配字符"n"。"\n"匹配换行符。序列"\\\\"匹配"\\"，"\\("匹配"("。 |
| ^ | 匹配输入字符串开始的位置。如果设置了 **RegExp** 对象的 **Multiline** 属性，^ 还会与"\n"或"\r"之后的位置匹配。 |
| $ | 匹配输入字符串结尾的位置。如果设置了 **RegExp** 对象的 **Multiline** 属性，$ 还会与"\n"或"\r"之前的位置匹配。 |
| \* | 零次或多次匹配前面的字符或子表达式。例如，zo\* 匹配"z"和"zoo"。\* 等效于 {0,}。 |
| + | 一次或多次匹配前面的字符或子表达式。例如，"zo+"与"zo"和"zoo"匹配，但与"z"不匹配。+ 等效于 {1,}。 |
| ? | 零次或一次匹配前面的字符或子表达式。例如，"do(es)?"匹配"do"或"does"中的"do"。? 等效于 {0,1}。 |
| {*n*} | *n*是非负整数。正好匹配 *n* 次。例如，"o{2}"与"Bob"中的"o"不匹配，但与"food"中的两个"o"匹配。 |
| {*n*,} | *n*是非负整数。至少匹配 *n*次。例如，"o{2,}"不匹配"Bob"中的"o"，而匹配"foooood"中的所有 o。"o{1,}"等效于"o+"。"o{0,}"等效于"o\*"。 |
| {*n*,*m*} | *m* 和 *n* 是非负整数，其中 *n* <= *m*。匹配至少 *n* 次，至多 *m* 次。例如，"o{1,3}"匹配"fooooood"中的头三个 o。'o{0,1}' 等效于 'o?'。注意：您不能将空格插入逗号和数字之间。 |
| ? | 当此字符紧随任何其他限定符（\*、+、?、{*n*}、{*n*,}、{*n*,*m*}）之后时，匹配模式是"非贪心的"。"非贪心的"模式匹配搜索到的、尽可能短的字符串，而默认的"贪心的"模式匹配搜索到的、尽可能长的字符串。例如，在字符串"oooo"中，"o+?"只匹配单个"o"，而"o+"匹配所有"o"。 |
| . | 匹配除"\r\n"之外的任何单个字符。若要匹配包括"\r\n"在内的任意字符，请使用诸如"[\s\S]"之类的模式。 |
| (*pattern*) | 匹配 *pattern* 并捕获该匹配的子表达式。可以使用 **$0…$9** 属性从结果"匹配"集合中检索捕获的匹配。若要匹配括号字符 ( )，请使用"\("或者"\)"。 |
| (?:*pattern*) | 匹配 *pattern* 但不捕获该匹配的子表达式，即它是一个非捕获匹配，不存储供以后使用的匹配。这对于用"or"字符 (|) 组合模式部件的情况很有用。例如，'industr(?:y|ies) 是比 'industry|industries' 更经济的表达式。 |
| (?=*pattern*) | 执行正向预测先行搜索的子表达式，该表达式匹配处于匹配 *pattern* 的字符串的起始点的字符串。它是一个非捕获匹配，即不能捕获供以后使用的匹配。例如，'Windows (?=95|98|NT|2000)' 匹配"Windows 2000"中的"Windows"，但不匹配"Windows 3.1"中的"Windows"。预测先行不占用字符，即发生匹配后，下一匹配的搜索紧随上一匹配之后，而不是在组成预测先行的字符后。 |
| (?!*pattern*) | 执行反向预测先行搜索的子表达式，该表达式匹配不处于匹配 *pattern* 的字符串的起始点的搜索字符串。它是一个非捕获匹配，即不能捕获供以后使用的匹配。例如，'Windows (?!95|98|NT|2000)' 匹配"Windows 3.1"中的 "Windows"，但不匹配"Windows 2000"中的"Windows"。预测先行不占用字符，即发生匹配后，下一匹配的搜索紧随上一匹配之后，而不是在组成预测先行的字符后。 |
| *x*|*y* | 匹配 *x* 或 *y*。例如，'z|food' 匹配"z"或"food"。'(z|f)ood' 匹配"zood"或"food"。 |
| [*xyz*] | 字符集。匹配包含的任一字符。例如，"[abc]"匹配"plain"中的"a"。 |
| [^*xyz*] | 反向字符集。匹配未包含的任何字符。例如，"[^abc]"匹配"plain"中"p"，"l"，"i"，"n"。 |
| [*a-z*] | 字符范围。匹配指定范围内的任何字符。例如，"[a-z]"匹配"a"到"z"范围内的任何小写字母。 |
| [^*a-z*] | 反向范围字符。匹配不在指定的范围内的任何字符。例如，"[^a-z]"匹配任何不在"a"到"z"范围内的任何字符。 |
| \b | 匹配一个字边界，即字与空格间的位置。例如，"er\b"匹配"never"中的"er"，但不匹配"verb"中的"er"。 |
| \B | 非字边界匹配。"er\B"匹配"verb"中的"er"，但不匹配"never"中的"er"。 |
| \c*x* | 匹配 *x* 指示的控制字符。例如，\cM 匹配 Control-M 或回车符。*x* 的值必须在 A-Z 或 a-z 之间。如果不是这样，则假定 c 就是"c"字符本身。 |
| \d | 数字字符匹配。等效于 [0-9]。 |
| \D | 非数字字符匹配。等效于 [^0-9]。 |
| \f | 换页符匹配。等效于 \x0c 和 \cL。 |
| \n | 换行符匹配。等效于 \x0a 和 \cJ。 |
| \r | 匹配一个回车符。等效于 \x0d 和 \cM。 |
| \s | 匹配任何空白字符，包括空格、制表符、换页符等。与 [ \f\n\r\t\v] 等效。 |
| \S | 匹配任何非空白字符。与 [^ \f\n\r\t\v] 等效。 |
| \t | 制表符匹配。与 \x09 和 \cI 等效。 |
| \v | 垂直制表符匹配。与 \x0b 和 \cK 等效。 |
| \w | 匹配任何字类字符，包括下划线。与"[A-Za-z0-9\_]"等效。 |
| \W | 与任何非单词字符匹配。与"[^A-Za-z0-9\_]"等效。 |
| \x*n* | 匹配 *n*，此处的 *n* 是一个十六进制转义码。十六进制转义码必须正好是两位数长。例如，"\x41"匹配"A"。"\x041"与"\x04"&"1"等效。允许在正则表达式中使用 ASCII 代码。 |
| \*num* | 匹配 *num*，此处的 *num* 是一个正整数。到捕获匹配的反向引用。例如，"(.)\1"匹配两个连续的相同字符。 |
| \*n* | 标识一个八进制转义码或反向引用。如果 \*n* 前面至少有 *n* 个捕获子表达式，那么 *n* 是反向引用。否则，如果 *n* 是八进制数 (0-7)，那么 *n*是八进制转义码。 |
| \*nm* | 标识一个八进制转义码或反向引用。如果 \*nm* 前面至少有 *nm* 个捕获子表达式，那么 *nm* 是反向引用。如果 \*nm* 前面至少有 *n* 个捕获，则 *n* 是反向引用，后面跟有字符 *m*。如果两种前面的情况都不存在，则 \*nm* 匹配八进制值 *nm*，其中 *n*和 *m* 是八进制数字 (0-7)。 |
| \nml | 当 *n* 是八进制数 (0-3)，*m* 和 *l* 是八进制数 (0-7) 时，匹配八进制转义码 *nml*。 |
| \u*n* | 匹配 *n*，其中 *n* 是以四位十六进制数表示的 Unicode 字符。例如，\u00A9 匹配版权符号 (©)。 |