regexp包及正则表达式的应用

目录：

一、 正则表达式

（一）、概述：

1、概念：

正则表达式(regular expression)就是由元字符组成的一种字符串匹配的模式，使用这种模式可以实现对文本内容解析、校验、替换。

2、正则表达式的用途：

1、数据有效性验证：用户注册模块是应用正则表达式最集中的地方，主要是用于验证用户帐号、密码、EMAIL、电话号码、QQ号码、身份证号码、家庭地址等信息。如果填写的内容与正则表达式不匹配，可以断定填写的内容是不合乎要求或虚假的信息，那么在将表单提交到服务器进一步处理前，JavaScript程序会检查表单以确认用户填写的是有效信息。采用正则表达式会使得数据校验的工作量大大减轻。

2、模糊查询，批量替换。可以在文档中使用一个正则表达式来查找匹配的特定文字，然后可以全部将其删除，或者替换为别的文字。

（二）、正则表达式中主要元字符：【其中常用的元字符用红色标出，红色的元字符必须掌握。难点用蓝色标出，难点在一般的应用中并不常用】

1.

\ 将下一个字符标记为一个特殊字符、或一个原义字符、或一个 向后引用、或一个八进制转义符。例如，'n'

匹配字符 "n"。'\n' （newline）匹配一个换行符。序列 '\\' 匹配 "\" 而

"\(" 则匹配 "("。 ‘\r’ （return）

2.

^ 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。

3.

$ 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。

4.

* 匹配前面的子表达式零次或多次。例如，zo* 能匹配

"z" 以及 "zoo"。* 等价于{0,}。

5.

+ 匹配前面的子表达式一次或多次。例如，'zo+' 能匹配 "zo" 以及

"zoo"，但不能匹配 "z"。+ 等价于 {1,}。

6.

? 匹配前面的子表达式零次或一次。例如，"do(es)?" 可以匹配 "do" 或

"does" 中的"do" 。? 等价于 {0,1}。

7.

{n} n 是一个非负整数。匹配确定的 n 次。例如，'o{2}' 不能匹配

"Bob" 中的 'o'，但是能匹配 "food" 中的两个 正好匹配的个数

o。

8.

{n,} n 是一个非负整数。至少匹配n 次。例如，'o{2,}' 不能匹配

"Bob" 中的 'o'，但能匹配 "foooood" 中的所有

o。'o{1,}' 等价于 'o+'。'o{0,}' 则等价于

'o*'。

9.

{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}" 将匹配 "fooooood" 中的前三个
o。'o{0,1}' 等价于 'o?'。请注意在逗号和两个数之间不能有空格。
10.
? 当该字符紧跟在任何一个其他限制符 (*, +, ?, {n}, {n,}, {n,m})
后面时，匹配模式是非贪婪的。*非贪婪模式*尽可能少的匹配所搜索的字符
串，而默认的*贪婪模式*则尽可能多的匹配所搜索的字符串。例如，对于字符
串
"oooo"，'o+?' 将匹配单个 "o"，而 'o+'
将匹配所有 'o'。
11.
点“. ” 匹配除 "\n"之外的任何单个字符。要匹配包括 '\n'
在内的任何字符，请使用象 '[.\n]' 的模式。
12.
x|y 匹配 x 或 y。“|”代表“或”的意思。例如，'z|food'
能匹配 "z" 或 "food"。'(z|f)ood' 则匹配
"zood" 或 "food"。
13.
[xyz] 字符集合。匹配所包含的任意一个字符。例如， '[abc]' 可以匹配
"plain" 中的 'a'。
14.
[^xyz] 负值字符集合。匹配未包含的任意字符。例如， '[^abc]' 可以匹配
"plain" 中的'p'。
15.
[a-z] 字符范围。匹配指定范围内的任意字符。例如，'[a-z]' 可以匹配 'a'
到 'z' 范围内的任意小写字母字符。
16. [^a-z]
负值字符范围。匹配任何不在指定范围内的任意字符。例如，'[^a-z]' 可以匹
配任何不在 'a' 到
'z' 范围内的任意字符。
17. \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'。
18. \B

匹配非单词边界。'er\B' 能匹配 "verb" 中的 'er'，但不能匹配
"never" 中的 'er'。
19. \cx
匹配由 x 指明的控制字符。例如， \cM 匹配一个 Control-M 或回车符。x 的
值必须为 A-Z 或 a-z 之一。否则，将
c 视为一个原义的 'c' 字符。
20.
\d 匹配一个数字。等价于 [0-9]。 digital
21.
\D 匹配一个非数字。等价于 [^0-9]。
22. \f
匹配一个换页符。等价于 \x0c 和 \cL。
23.
\n 匹配一个换行符。等价于 \x0a 和 \cJ。
24.
\r 匹配一个回车符。等价于 \x0d 和 \cM。
25.
\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]。
（space）
26. \S
匹配任何非空白字符。等价于 [^ \f\n\r\t\v]。
27. \t
匹配一个制表符。等价于 \x09 和 \cI。
28. \v
匹配一个垂直制表符。等价于 \x0b 和 \cK。
29.
\w 匹配包括下划线的任何单词字符。等价于'[A-Za-z0-9_]'。 word
30. \W
匹配任何非单词字符。等价于 '[^A-Za-z0-9_]'。
31. \num
匹配 num，其中 num 是一个正整数。对所获取的匹配的引用。例如，'(.)\1'
匹配两个连续的相同字符。
32. \xn
匹配 n，其中 n 为十六进制转义值。十六进制转义值必须为确定的两个数字

长。例如，'\x41' 匹配
"A"。'\x041' 则等价于 '\x04' &
"1"。正则表达式中可以使用 ASCII 编码。
33. \un
匹配 n，其中 n 是一个用四个十六进制数字表示的 Unicode 字符。例如，
\u00A9 匹配版权符号
(?)。
34. (pattern)
匹配 括号内pattern所代表的表达式。是成组匹配。
35. (?=pattern) *正向预查*。例如windows(?=95/98/2000/NT)，含义是匹配“windows”后面可以是“95”“98”“2000”或者“NT”。
36. (?!pattern) *负向预查*。例windows(?!95/98)，含义是匹配“windows”后面不是“95”或“98”的其它字符串。

【特别备注】正则表达式的备注说明

1、大写英文字母的正则表达式，除了可以写成[A-Z]，还可以写成[\x41-\x5A]。因为在ASCII码字典中A-Z被排在了65－90号（也就是ASCII码的第66到第91位），换算成16进制就是0x41-0x5A；

2、[0-9]，可以写成[\x30-\x39]；

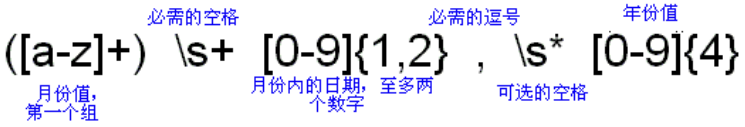
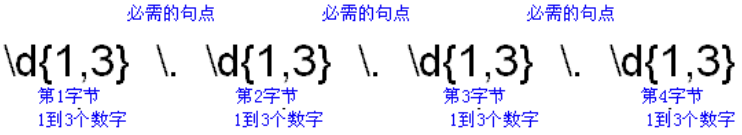
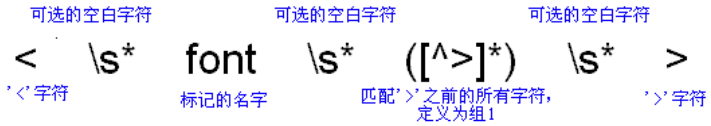
3、[a-z]，可以写成[\x61-\x7A]。

4、[A-Z]，可以写成[\x41-\x5A]。

4、中文的正则表达式为：[\u4E00-\u9FA5]

因为中文在unicode编码字典中排在4E00到9FA5之间。换成10进制，也就是第19968号到40869号是中文字，一共20902个中文字被搜录到unicode编码集中。(常识了解：第19968号是“一”，而第40869号是“龥”——发音为yu)。

（三）、图解正则表达式：

	匹配所有Moth DD,YY
	匹配所有Month DD,Y 月份值为第一个组
	匹配所有123-12-1234
	匹配IP地址 \d{1,3}\.\d{1,3}\.
	匹配FONT标记的所有 匹配这些字符本身，

1. ^ $
2. * + ?
3. { } ( ) [ ]
4. \ / . |

【备注：】以上特殊符号在实际定义regexp字符串的时候，实际上使用两个反斜杠"\\"。

（五）、元字符优先级顺序（从高到低，从左到右）

1. \ 转义字符
2. () 圆括号， [] 方括号
3. * + ? {n} {n,} {n,m} 限定符
4. ^ $ 开始和结束标识
5. | "或"操作

（六）、常用的正则表达式的写法：

1、中文字符：^[\u4E00-\u9FA5]+$

2、手机号码：^(86)?0?1\d{10}$

电话号码：^((d{3,4})|d{3,4}-)?d{7,8}$

3、Email地址：^[\w-]+[\w-.]?@[\w-]+(\.[A-Za-z]{2,5})+$

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

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

4、URL网址：^http://([w-]+.)+[w-]+(([w-.)?%&=]*)?$

URL网址：http://([w-]+.)+[w-]+(/[w- ./?%&=]*)?

5、密码（安全级别中）：^(\d+[A-Za-z]\w*|[A-Za-z]+\d\w*)$

6、密码（安全级别高）：^(\d+[a-zA-Z~!@#$%^&(){}][\w~!@#$%^&(){}]*|[a-zA-Z~!@#$%^&(){}]+\d[\w~!@#$%^&(){}]*)$

【备注：】对于同一个需求的正则表达式，因理解不同和验证的严格程度不同而差异很大，没有固定的统一写法。只要尽量与需求进行匹配就可以。

二、正则表达式在Go中的用法

Go语言中的正则表达式采用RE2语法（除了\c、\C），和Perl、Python等语言的正则基本一致。

（一）、Regexp结构体

（三）、

三、常用正则表达式的写法：

（一）、【附录1：】常用的正则表达式写法一：

匹配特定数字：

```
^[1-9]d*$        //匹配正整数
^-[1-9]d*$     //匹配负整数
```

^-?[1-9]d*$　　//匹配整数
^[1-9]d*|0$　//匹配非负整数（正整数 + 0）
^-[1-9]d*|0$　　//匹配非正整数（负整数 + 0）
^[1-9]d*.d*|0.d*[1-9]d*$　　//匹配正浮点数
^-([1-9]d*.d*|0.d*[1-9]d*)$　//匹配负浮点数
^-?([1-9]d*.d*|0.d*[1-9]d*|0?.0+|0)$　//匹配浮点数
^[1-9]d*.d*|0.d*[1-9]d*|0?.0+|0$　　//匹配非负浮点数（正浮点数 + 0）
^(-([1-9]d*.d*|0.d*[1-9]d*))|0?.0+|0$　　//匹配非正浮点数（负浮点数 + 0）

匹配特定字符串：
^[A-Za-z]+$　　//匹配由26个英文字母组成的字符串
^[A-Z]+$　　//匹配由26个英文字母的大写组成的字符串
^[a-z]+$　　//匹配由26个英文字母的小写组成的字符串
^[A-Za-z0-9]+$　　//匹配由数字和26个英文字母组成的字符串
^w+$　　//匹配由数字、26个英文字母或者下划线组成的字符串

只能输入数字：“^[0-9]*$”
只能输入n位的数字：“^d{n}$”
只能输入至少n位数字：“^d{n,}$”
只能输入m-n位的数字：“^d{m,n}$”
只能输入零和非零开头的数字：“^(0|[1-9][0-9]*)$”
只能输入有两位小数的正实数：“^[0-9]+(.[0-9]{2})?$”
只能输入有1-3位小数的正实数：“^[0-9]+(.[0-9]{1,3})?$”
只能输入非零的正整数：“^+?[1-9][0-9]*$”
只能输入非零的负整数：“^-[1-9][0-9]*$”
只能输入长度为3的字符：“^.{3}$”
只能输入由26个英文字母组成的字符串：“^[A-Za-z]+$”
只能输入由26个大写英文字母组成的字符串：“^[A-Z]+$”
只能输入由26个小写英文字母组成的字符串：“^[a-z]+$”
只能输入由数字和26个英文字母组成的字符串：“^[A-Za-z0-9]+$”
只能输入由数字、26个英文字母或者下划线组成的字符串：“^w+$”
验证用户密码:“^[a-zA-Z]w{5,17}$”正确格式为：以字母开头，长度在6-18之间，

只能包含字符、数字和下划线。
验证是否含有^%&’,;=?$”等字符：“[^%&’,;=?$x22]+”
只能输入汉字：“^[u4e00-u9fa5]{0,}$”
验证身份证号（15位或18位数字）：“^d{15}|d{}18$”
验证一年的12个月：“^(0?[1-9]|1[0-2])$”正确格式为：“01”-“09”和“1”“12”
验证一个月的31天：“^((0?[1-9])|((1|2)[0-9])|30|31)$”

匹配中文字符的正则表达式： [u4e00-u9fa5]
匹配双字节字符(包括汉字在内)：[^x00-xff]
匹配空行的正则表达式：n[s|]*r
匹配HTML标记的正则表达式：/< (.*)>.*|< (.*) />/
匹配首尾空格的正则表达式：(^s*)|(s*$)

（二）、【附录2：】常用的正则表达式写法二：

1.
整数或者小数：^[0-9]+\.{0,1}[0-9]{0,2}$
2.
只能输入数字："^[0-9]*$"。
3.
只能输入n位的数字："^\d{n}$"。
4.
只能输入至少n位的数字："^\d{n,}$"。
5.
只能输入m~n位的数字：。"^\d{m,n}$"
6.
只能输入零和非零开头的数字："^(0|[1-9][0-9]*)$"。
7.
只能输入有两位小数的正实数："^[0-9]+(.[0-9]{2})?$"。
8.
只能输入有1~3位小数的正实数："^[0-9]+(.[0-9]{1,3})?$"。
9.
只能输入非零的正整数："^\+?[1-9][0-9]*$"。
10.
只能输入非零的负整数："^\-[1-9][]0-9"*$。
11.
只能输入长度为3的字符："^.{3}$"。
12.
只能输入由26个英文字母组成的字符串："^[A-Za-z]+$"。
13.
只能输入由26个大写英文字母组成的字符串："^[A-Z]+$"。
14.
只能输入由26个小写英文字母组成的字符串："^[a-z]+$"。
15.

只能输入由数字和26个英文字母组成的字符串："^[A-Za-z0-9]+$"。

16.

只能输入由数字、26个英文字母或者下划线组成的字符串："^\w+$"。

17.

验证用户密码："^[a-zA-Z]\w{5,17}$"正确格式为：以字母开头，长度在6~18之间，只能包含字符、数字和下划线。

18.

验证是否含有^%&',;=?$\"等字符："[^%&',;=?$\x22]+"。

19.

只能输入汉字："^[\u4e00-\u9fa5]{0,}$"

20.

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

21.

验证InternetURL："^http://([\w-]+\.)+[\w-]+(/[\w-./?%&=]*)?$"。

22.

验证电话号码："^(\(\d{3,4}-)|\d{3.4}-)?\d{7,8}$"

正确格式为："XXX-XXXXXXX"、"XXXX-XXXXXXXX"、"XXX-XXXXXXX"、"XXX-XXXXXXXX"、"XXXXXXX"和"XXXXXXXX"。

23.

验证身份证号（15位或18位数字）："^\d{15}|\d{18}$"。

24.

验证一年的12个月："^(0?[1-9]|1[0-2])$"正确格式为："01"～"09"和"1"～"12"。

25.

验证一个月的31天："^((0?[1-9])|((1|2)[0-9])|30|31)$"正确格式为；"01"～"09"和"1"～"31"。

26.

匹配中文字符的正则表达式： [\u4e00-\u9fa5]

27.

匹配双字节字符(包括汉字在内)：[^\x00-\xff]

28.

匹配空行的正则表达式：\n[\s|]*\r

29.

匹配html标签的正则表达式：<(.*)>(.*)<\/(.*)>|<(.*)\/>

30.

匹配首尾空格的正则表达式：(^\s*)|(\s*$)

【备注】

URL的组成部分?

- 协议　http://
- 主机名　localhost
- 端口　:3000
  - 由于物理端口和逻辑端口数量较多，为了对端口进行区分，将每个端口进行了编号，这就是端口号。我们主要研究的是逻辑端口号.我们平时所说的端口号也是指的逻辑端口号。
  - 逻辑端口是指逻辑意义上用于区分服务的端口，一台服务器有256*256个端口，端口号的范围从0到65535。
  - 0-1023是公认端口号，即已经公认定义或为将要公认定义的软件保留的。
    - HTTP传输是80端口
    - HTTPS传输是443端口
    - FTP服务是21端口
  - 1024-65535是并没有公共定义的端口号，用户可以自己定义这些端口的作用。一般都使用>1023的端口。
- 路径　/search
- 查询参数　?wd=bitcoin
- 信息片段（散列、锚点）　#history