**正则表达式**

#### 正则表达式---基础

1. []的使用

[ab]代表有一个a或者b的字符串，等同于a|b

1. 数字

[0-9]代表0-9的数字，等同于\d

1. 英文字符

[a-z]代表所有小写英文字符

[A-Z]代表所有大写英文字符

[a-zA-Z]代表所有英文字符

1. 空格字符

\t--------匹配一个制表符

预定义字符的反义：预定义字符的大写形式都是小写的反义

\D---->1位非数字字符

\v------匹配一个垂直制表符

\n------匹配一个换行符

\r-------匹配一个回车符

1. \w与\W

\w代表所有的大小写英文字符、数字、下划线，等同于[0-9a-zA-Z\_]

\W代表非\w即不是大小写字符、数字、下划线的字符，等同于[^0-9a-zA-Z\_]

1. \s与\S

\s代表所有的空格符，包括空格、制表符、换行符等，等同于[ \f\n\r\t\v]

\S代表所有的非空格符，等同于[^\f\n\r\t\v]

1. .

‘.’ 代表除了回车外的任意一个字符，.{3}代表除回车外的任意三个字符

1. {n,m}的使用

{n,m}代表出现了n到m次，[a-z]{1,3}代表出现1到3次的小写英文字符

{n,}代表至少出现n次，多了不限

{n}代表出现了n次

1. \*，+，?

\*代表出现0次或多次，等价于{0,}

+代表出现1次或多次，等价于{1,}

?代表出现0次或1次，等价于{0,1}

1. ^与$

^在[]中使用代表非，在[]外使用代表以某某开头

$代表以某某结尾

1. 贪婪模式与懒惰模式

贪婪模式：默认情况下，正则表达式会匹配最大的符合条件的字符串, \*、+和?限定符都是贪婪的，因为它们会尽可能多的匹配文字，只有在它们的后面加上一个?就可以实现非贪婪或最小匹配。

贪婪模式原因：(.\*) (.+)

懒惰模式：正则表达式仅匹配最小的符合规则的字符串

比如：筛选网页中的a元素：

<a\s+(.\*)href\s\*=\s\*["']([^'"]\*)["']

贪婪模式-->懒惰模式：.\*?

1. |

|代表‘或‘，(ab)|(bc)代表ab或者bc

#### 正则表达式---实例

#### 正则表达式---java

Pattern ***PATTERN\_TIMESTAMP*** = Pattern.*compile*(**"\\d{4}[-]\\d{2}[-]\\d{2}[ ]\\d{2}[:]\\d{2}[:]\\d{2}"**);

Matcher matcherTimestamp = ***PATTERN\_TIMESTAMP***.matcher(oldMessage);  
**if** (matcherTimestamp.find()) {  
 timestamp = matcherTimestamp.group();  
}