1.什么是模板引擎

预处理器，一个字符串，有些未处理的字符串，通过数据匹配来生成html。

2.正则表达式

普通字符和特殊字符组成的字符表达式

Pattern pattern = Pattern.compile("^o{2,5}");

Matcher mathcer = pattern.matcher("123oooe");

mathcer.group();//返回只与pattern匹配的内容，2-5个o将被返回，同理o{0,1}等价于o?

特殊字符

.任意字符，除了换行符\n

\* >=0个字符，例如 z\* = z{0,} 0个z或者无数个z

+ >=1个字符，例如 z\* = z{1,} 1个z或者无数个z

这里需要注意 . \* +都是贪心表达式，而在后面添加?可以转为“非贪心”表达式

定位符

^和$分别指字符串的开始与结束，\b描述单词的前或后边界，\B表示非单词边界。

不能^\*这样使用，限定符和定位符不能一起使用。

例如：捕获uri 正则：(\w+):\/\/([^/:]+)(:\d\*)?([^# ]\*) 字符串：http://www.google.com/test/test.html

\*\*关于\w的使用

正则表达式中\w不等效于[A-Za-z0-9]

"аdmin",注意到那个а了没?那是一个俄文字母,在会员注册过滤中使用\w验证并不能防止。

也就是说\w包括了英文字母以及俄文字母等，在会员注册这块我们肯定不希望类同的账号出现，要不然很容易造成会员间的误会。所以改为^[A-Za-z0-9\u4E00-\u9FA5-]{2,16}$验证，只允许数字、26个英文字母大小写、中文及其组合。

\*\*\*选择

?=

举例: Pattern pattern = Pattern.compile(“\\d{3}(?=a)”); 能匹配到的字符串是，连续三个数字之后出现了一个字母a，比如：111a,

Matcher.group()之后会是111，而不显示a,会被忽略。

?:

作用：清除由于使用圆括号产生的缓存在圆括号的后面加？:

举例： Pattern pat = Pattern.compile(“\\w{3}(?:a)”); 能匹配的字符串是三个随机的字符后跟一个a，

比如 Matcher matcher = pat.matcher(“1waa”);

Matcher.group()返回的是1waa

?!

作用：查找第一个不符合？！后面的模式的代码返回

举例：Pattern pattern = Pattern.compile(“\\w{3}(?!abc)”);

Matcher matcher = pattern.matcher(“123abca”);

Matcher.group();//输出是23a首先要不满足abc之前有任意的三个字符，那么123abc满足，所以往后一格，23abca,最后三个是bca，所以输出23a

<http://www.runoob.com/regexp/regexp-example.html> 1117

几个有用的正则表达式：

/\b([a-z]+) \1\b/gi 一个单词连续出现的位置。

/(\w+):\/\/([^/:]+)(:\d\*)?([^# ]\*)/ 将一个URL解析为协议、域、端口及相对路径。

/^(?:Chapter|Section) [1-9][0-9]{0,1}$/ 定位章节的位置。

/[-a-z]/ a至z共26个字母再加一个-号。

/ter\b/ 可匹配chapter，而不能匹配terminal。

/\Bapt/ 可匹配chapter，而不能匹配aptitude。

/Windows(?=95 |98 |NT )/ 可匹配Windows95或Windows98或WindowsNT，当找到一个匹配后，从Windows后面开始进行下一次的检索匹配。

/^\s\*$/ 匹配空行。

/\d{2}-\d{5}/ 验证由两位数字、一个连字符再加 5 位数字组成的 ID 号。

/<\s\*(\S+)(\s[^>]\*)?>[\s\S]\*<\s\*\/\1\s\*>/ 匹配 HTML 标记。

3.json的使用

json语法

json

http://www.cnblogs.com/hustskyking/p/principle-of-javascript-template.html 模板引擎