2-2 正则表达式的总结

听完老师的讲解，小慕帮大家再次巩固正则内容：

一、什么是正则表达式

正则表达式是一个特殊的字符序列，便于检查一个字符串是否与某种模式匹配。

二、正则表达式的作用

在实际开发过程中经常会有查找符合某些复杂规则的字符串的需要，比如:手机号、邮箱、图片地址等，这时候想匹配或者查找符合某些规则的字符串就可以使用正则表达式了。

三、正则表达式的特点

- 正则表达式的语法太多，可读性差
- 正则表达式通用行很强，能够适用于很多编程语言

四、匹配单个字符

符号	描 述	示例
literal	匹配文本字符串的字面值 literal	foo
re1|re2	匹配正则表达式 re1 或者 re2	foo|bar
.	匹配任何字符（除了\n 之外）	b.b
^	匹配字符串起始部分	^Dear
$	匹配字符串终止部分	/bin/*sh$
*	匹配 0 次或者多次前面出现的正则表达式	[A-Za-z0-9]*
+	匹配 1 次或者多次前面出现的正则表达式	[a-z]+\.com
?	匹配 0 次或者 1 次前面出现的正则表达式	goo?
{N}	匹配 N 次前面出现的正则表达式	[0-9]{3}
{M,N}	匹配 M～N 次前面出现的正则表达式	[0-9]{5,9}
[…]	匹配来自字符集的任意单一字符	[aeiou]
[..x-y..]	匹配 x～y 范围中的任意单一字符	[0-9], [A-Za-z]
[^…]	不匹配此字符集中出现的任何一个字符，包括某一范围的字符（如果在此字符集中出现）	[^aeiou], [^A-Za-z0-9]
(*|+|?|{})?	用于匹配上面频繁出现/重复出现符号的非贪婪版本（*、+、 ?、 {}）	.*?[a-z]
(…)	匹配封闭的正则表达式，然后另存为子组	([0-9]{3})?,f(oo|u)bar

正则表达式中的特殊字符

特殊字符	描 述	示例
\d	匹配任何十进制数字，与[0-9]一致（\D 与\d 相反，不匹配任何非数值型的数字）	data\d+.txt
\w	匹配任何字母数字字符，与[A-Za-z0-9_]相同（\W 与之相反）	[A-Za-z_]\w+
\s	匹配任何空格字符，与[\n\t\r\v\f]相同（\S 与之相反）	of\sthe
\b	匹配任何单词边界（\B 与之相反）	\bThe\b
\N	匹配已保存的子组 N（参见上面的(…)）	price: \16
\c	逐字匹配任何特殊字符 c（即，仅按照字面意义匹配，不匹配特殊含义）	\., \\, *
\A(\Z)	匹配字符串的起始（结束）（另见上面介绍的^和$）	\ADear

正则表达式中的扩展表示法

扩展表示法	描 述	示例
(?iLmsux)	在正则表达式中嵌入一个或者多个特殊“标记” 参数（或者通过函数/方法）	(?x)， (? im)
(?:…)	表示一个匹配不用保存的分组	(?:\w+\.)*
(?P<name>…)	像一个仅由 name 标识而不是数字 ID 标识的正则分组匹配	(?P<data>)
(?P=name)	在同一字符串中匹配由(?P<name)分组的之前文本	(?P=data)
(?#…)	表示注释，所有内容都被忽略	(?#comment)
(?=…)	匹配条件是如果…出现在之后的位置，而不使用输入字符串；称作正向前视断言	(?=.com)
(?!…)	匹配条件是如果…不出现在之后的位置，而不使用输入字符串；称作负向前视断言	(?!.net)
(?<=…)	匹配条件是如果…出现在之前的位置，而不使用输入字符串；称作正向后视断言	(?<=800-)
(?<!…)	匹配条件是如果…不出现在之前的位置，而不使用输入字符串；称作负向后视断言	(?<!192\.168\.)
(?(id/name)Y|N)	如果分组所提供的 id 或者 name（名称）存在，就返回正则表达式的条件匹配 Y，如果不存在，就返回 N； |N 是可选项	(?(1)y|x

下一节