# 正则表达式

## 正则表达式的概念

**正则表达式**（英语：Regular Expression，在代码中常简写为regex）。

正则表达式是一个字符串，使用单个字符串来描述、用来定义匹配规则，匹配一系列符合某个句法规则的字符串。在开发中，正则表达式通常被用来检索、替换那些符合某个规则的文本。

## 正则表达式的匹配规则

参照帮助文档，在Pattern类中有正则表达式的的规则定义，正则表达式中明确区分大小写字母。我们来学习语法规则。

正则表达式的语法规则：

**字符：x**

含义：代表的是字符x

例如：匹配规则为 **"a"**，那么需要匹配的字符串内容就是 ”a”

**字符：\\**

含义：代表的是反斜线字符'\'

例如：匹配规则为**"\\" ，**那么需要匹配的字符串内容就是 ”\”

**字符：\t**

含义：制表符

例如：匹配规则为**"\t**" ，那么对应的效果就是产生一个制表符的空间

**字符：\n**

含义：换行符

例如：匹配规则为**"\n"**，那么对应的效果就是换行,光标在原有位置的下一行

**字符：\r**

含义：回车符

例如：匹配规则为**"\r"** ，那么对应的效果就是回车后的效果,光标来到下一行行首

**字符类：[abc]**

含义：代表的是字符a、b 或 c

例如：匹配规则为**"[abc]"** ，那么需要匹配的内容就是字符a，或者字符b，或字符c的一个

**字符类：[^abc]**

含义：代表的是除了 a、b 或 c以外的任何字符

例如：匹配规则为**"[^abc]"**，那么需要匹配的内容就是不是字符a，或者不是字符b，或不是字符c的任意一个字符

**字符类：[a-zA-Z]**

含义：代表的是a 到 z 或 A 到 Z，两头的字母包括在内

例如：匹配规则为**"[a-zA-Z]"**，那么需要匹配的是一个大写或者小写字母

**字符类：[0-9]**

含义：代表的是 0到9数字，两头的数字包括在内

例如：匹配规则为**"[0-9]"**，那么需要匹配的是一个数字

**字符类：[a-zA-Z\_0-9]**

含义：代表的字母或者数字或者下划线(即单词字符)

例如：匹配规则为**" [a-zA-Z\_0-9] "**，那么需要匹配的是一个字母或者是一个数字或一个下滑线

**预定义字符类：.**

含义：代表的是任何字符

例如：匹配规则为**" . "**，那么需要匹配的是一个任意字符。如果，就想使用 . 的话，使用匹配规则"\\."来实现

**预定义字符类：\d**

含义：代表的是 0到9数字，两头的数字包括在内，相当于[0-9]

例如：匹配规则为**"\d "**，那么需要匹配的是一个数字

**预定义字符类：\w**

含义：代表的字母或者数字或者下划线(即单词字符)，相当于**[a-zA-Z\_0-9]**

例如：匹配规则为**"\w "**，，那么需要匹配的是一个字母或者是一个数字或一个下滑线

**边界匹配器：^**

含义：代表的是行的开头

例如：匹配规则为**^[abc][0-9]$** ，那么需要匹配的内容从[abc]这个位置开始, 相当于左双引号

**边界匹配器：$**

含义：代表的是行的结尾

例如：匹配规则为**^[abc][0-9]$** ，那么需要匹配的内容以[0-9]这个结束, 相当于右双引号

**边界匹配器：\b**

含义：代表的是单词边界

例如：匹配规则为**"\b[abc]\b"** ，那么代表的是字母a或b或c的左右两边需要的是非单词字符(**[a-zA-Z\_0-9]**)

**数量词：X?**

含义：代表的是X出现一次或一次也没有

例如：匹配规则为**"a?"**，那么需要匹配的内容是一个字符a，或者一个a都没有

**数量词：X\***

含义：代表的是X出现零次或多次

例如：匹配规则为**"a\*"** ，那么需要匹配的内容是多个字符a，或者一个a都没有

**数量词：X+**

含义：代表的是X出现一次或多次

例如：匹配规则为**"a+"**，那么需要匹配的内容是多个字符a，或者一个a

**数量词：X{n}**

含义：代表的是X出现恰好 n 次

例如：匹配规则为**"a{5}"**，那么需要匹配的内容是5个字符a

**数量词：X{n,}**

含义：代表的是X出现至少 n 次

例如：匹配规则为**"a{5, }"**，那么需要匹配的内容是最少有5个字符a

**数量词：X{n,m}**

含义：代表的是X出现至少 n 次，但是不超过 m 次

例如：匹配规则为**"a{5,8}"**，那么需要匹配的内容是有5个字符a 到 8个字符a之间

规则："[0-9]{6,12}"

该规则需要匹配的内容是：**长度为6位到12位的数字。**

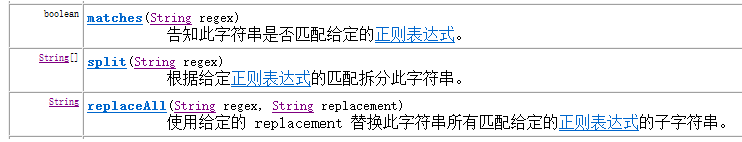
规则："1[34578][0-9]{9}"

该规则需要匹配的内容是：**11位的手机号码，第1位为1，第2位为3、4、5、7、8中的一个，后面9位为0到9之间的任意数字**。

规则："a\*b"

该规则需要匹配的内容是：**在多个a或零个a后面有个b；b必须为最后一个字符。**

## 字符串类中涉及正则表达式的常用方法



* public boolean **matches**(String regex) //判断字符串是否匹配给定的规则

举例：校验qq号码.

1: 要求必须是5-15位数字

2: 0不能开头

代码演示：

String qq = "604154942";

String regex = "[1-9][0-9]{4,14}";

**boolean** flag2 = qq.matches(regex);

举例：校验手机号码

1：要求为11位数字

2：第1位为1，第2位为3、4、5、7、8中的一个，后面9位为0到9之间的任意数字。

代码演示：

String phone = "18800022116";

String regex = "1[34578][0-9]{9}";

**boolean** flag = phone.matches(regex);

* public String[] **split**(String regex) //根据给定正则表达式的匹配规则，拆分此字符串

举例：分割出字符串中的的数字

代码演示：

String s = "18-22-40-65";

String regex = "-";

String[] result = s.split(regex);

代码演示：

String s = "18 22 40 65";

String regex = " ";

String[] result = s.split(regex);

* public String **replaceAll**(String regex,String replacement) //将符合规则的字符串内容，全部替换为新字符串

举例：把文字中的数字替换成\*

代码演示：

String s = "Hello12345World6789012";

String regex = "[0-9]";

String result = s.replaceAll(regex, "\*");

## 正则表达式练习

* 匹配正确的数字

匹配规则：

匹配正整数：”\\d+”

匹配正小数：”\\d+\\.\\d+”

匹配负整数：”-\\d+”

匹配负小数：”-\\d+\\.\\d+”

匹配保留两位小数的正数：”\\d+\\.\\d{2}”

匹配保留1-3位小数的正数：”\\d+\\.\\d{1,3}”

* 匹配合法的邮箱

匹配规则：

”[a-zA-Z\_0-9]+@[a-zA-Z\_0-9]+(\\.[a-zA-Z\_0-9]+)+”

”\\w+@\\w+(\\.\\w+)+”

* 获取IP地址(192.168.1.100)中的每段数字

匹配规则：

”\\.”