**java.util.regex 包**

java.util.regex 包是 Java 标准库中用于支持正则表达式操作的包。

java.util.regex 包主要包括以下三个类：

Pattern 类：

pattern 对象是一个正则表达式的编译表示。Pattern 类没有公共构造方法。要创建一个 Pattern 对象，你必须首先调用其公共静态编译方法，它返回一个 Pattern 对象。该方法接受一个正则表达式作为它的第一个参数。

Matcher 类：

Matcher 对象是对输入字符串进行解释和匹配操作的引擎。与Pattern 类一样，Matcher 也没有公共构造方法。你需要调用 Pattern 对象的 matcher 方法来获得一个 Matcher 对象。

PatternSyntaxException：

PatternSyntaxException 是一个非强制异常类，它表示一个正则表达式模式中的语法错误。

以下实例中使用了正则表达式 .\*runoob.\* 用于查找字符串中是否包了 runoob 子串：

实例

import java.util.regex.\*;

class RegexExample1{

public static void main(String[] args){

String content = "I am noob " +

"from runoob.com.";

String pattern = ".\*runoob.\*";

boolean isMatch = Pattern.matches(pattern, content);

System.out.println("字符串中是否包含了 'runoob' 子字符串? " + isMatch);

}

}

实例输出结果为：

字符串中是否包含了 'runoob' 子字符串? true

**捕获组**

捕获组是把多个字符当一个单独单元进行处理的方法，它通过对括号内的字符分组来创建。

例如，正则表达式 (dog) 创建了单一分组，组里包含"d"，"o"，和"g"。

捕获组是通过从左至右计算其开括号来编号。例如，在表达式（（A）（B（C））），有四个这样的组：

((A)(B(C)))

(A)

(B(C))

(C)

可以通过调用 matcher 对象的 groupCount 方法来查看表达式有多少个分组。groupCount 方法返回一个 int 值，表示matcher对象当前有多个捕获组。

还有一个特殊的组（group(0)），它总是代表整个表达式。该组不包括在 groupCount 的返回值中。

实例

下面的例子说明如何从一个给定的字符串中找到数字串：

RegexMatches.java 文件代码：

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class RegexMatches

{

public static void main( String[] args ){

// 按指定模式在字符串查找

String line = "This order was placed for QT3000! OK?";

String pattern = "(\\D\*)(\\d+)(.\*)";

// 创建 Pattern 对象

Pattern r = Pattern.compile(pattern);

// 现在创建 matcher 对象

Matcher m = r.matcher(line);

if (m.find( )) {

System.out.println("Found value: " + m.group(0) );

System.out.println("Found value: " + m.group(1) );

System.out.println("Found value: " + m.group(2) );

System.out.println("Found value: " + m.group(3) );

} else {

System.out.println("NO MATCH");

}

}

}

以上实例编译运行结果如下：

Found value: This order was placed for QT3000! OK?

Found value: This order was placed for QT

Found value: 3000

Found value: ! OK?