

Java ???(Regular Expression)

1. ??

正则表达式 (Regular Expression) 是一种用于匹配字符串中模式的强大工具。在 Java 中，正则表达式通过 `java.util.regex` 包中的类来实现。

1.1. 正则表达式的基本语法

正则表达式的基本语法包括元字符、转义字符、量词、分组和断言等。以下是一些基本的元字符和转义字符：

- `[]`：字符类，用于匹配括号内的任意一个字符。
 - `[a-z]`：匹配任意小写字母。
- `{n,m}`：量词，用于匹配括号内的模式 `n` 到 `m` 次。
 - `a{2,4}`：匹配 'a' 2 到 4 次。
- `^`, `$`, `.`, `|`, `\`：特殊元字符。
 - `^abc`：匹配以 'abc' 开头的字符串。

1.2. 正则表达式中的特殊字符

- `\d`：匹配任意数字 (0-9)。
- `\w`：匹配任意字母、数字、下划线 (a-zA-Z0-9_)。
- `\s`：匹配任意空白字符 (空格、制表符、换行符)。
- `\b`：匹配任意边界。

2. 正则表达式的应用

2.1. Pattern 类

`Pattern` 类是 `java.util.regex` 包中的一个类，用于编译正则表达式。它提供了 `compile()` 方法来编译正则表达式，并返回一个 `Pattern` 对象。

2.1.1. Pattern 类的常用方法

```
Pattern pattern = Pattern.compile("a*b");
```

编译正则表达式 "a*b" 时，'a' 表示 0 个或多个 'a'，'b' 表示 1 个或多个 'b'。

2.1.2. Matcher ??? ??

Matcher 是 Pattern 的子类，用于匹配字符串。

```
Pattern pattern = Pattern.compile("a*b");
Matcher matcher = pattern.matcher("aaab");
boolean matches = matcher.matches(); // true
```

2.2. Pattern? Matcher? ?? ???

- matches(): 是否匹配整个字符串
- find(): 是否匹配下一个子字符串
- replaceAll(): 替换所有匹配的字符串

3. ?????? ?? ??

3.1. ??? ?? ??

正则表达式用于验证电子邮件地址。

```
Pattern pattern = Pattern.compile("^([a-zA-Z0-9_+&*~]+(?:\\.[a-zA-Z0-9_+&*~]+)*)@(?:[a-zA-Z0-9-]+\\.)+[a-zA-Z]{2,7}$");
Matcher matcher = pattern.matcher("example@domain.com");
boolean isValid = matcher.matches(); // true
```

3.2. ????? ?? ??

正则表达式用于验证电话号码。

```
Pattern pattern = Pattern.compile("^\\d{3}-\\d{3,4}-\\d{4}$");
Matcher matcher = pattern.matcher("010-1234-5678");
boolean isValid = matcher.matches(); // true
```

Revision #1

Created 22 May 2025 05:15:02 by Dain

Updated 22 May 2025 05:16:16 by Dain