

Regular Expression

java.util.regex

- Pattern: 定義一個regular expression
- Matcher: 指定搜尋的字串

PatternExample.java

Character Classes

- .: 任意單一字元（字元、數字或特殊字元）
- [abc]: a、b或c
- [^abc]: 不是a、b或c
- [a-c]: 指定一個範圍的字元
- |: 或

Examples: It was the best of times

- w.s: It `was` the best of times
- w[abc]s: It `was` the best of times
- t[^aeo]mes: It was the best of `times`

CustomCharClassExamples.java

Predifined Character Classes

- \d: 數字, [0-9]
- \D: 不是數字, [^0-9]
- \w: 文字, [a-zA-Z0-9_]
- \W: 不是文字, [^a-zA-Z0-9_]
- \s: 空白, [\r\t\n\f\0XB]
- \S: 不是空白, [^\r\t\n\f\0XB]
 - \t: Tab
 - \n: New line
 - \r: Carriage return
 - \f: Form feed
 - \0XB: Vertical tab

Examples: Jo told me 20 ways to San Jose in 15 minutes.

- `\d\d`: Jo told me `20` ways to San Jose in 15 minutes.
- `\sin\s`: Jo told me 20 ways to San Jose `in` 15 minutes.
- `\Sin\S`: Jo told me 20 ways to San Jose in 15 `minu`tes.

PredefinedCharClassExample.java

Quantifiers

- `*`: 前面的字元出現零到多次
- `+`: 前面的字元出現一到多次
- `?`: 前面的字元必須出現一或零次
- `{n}`: 前面的字元必須出現n次
- `{m,n}`: 前面的字元必須出現m到n次
- `{m,}`: 前面的字元必須出現m或多次
- `(xx){n}`: 指定的一組字元出現n次

Examples: Longlonglong ago, in a galaxy far far away

- `ago.*`: Longlonglong `ago, in a galaxy far far away`
- `gal.{3}`: Longlonglong ago, in a `galaxy` far far away
- `(long){2}`: Long `longlong` ago, in a galaxy far far away

QuantifierExample.java

Greediness

Longlonglong ago, in a balaxy far far away.

- `ago.*far`: Longlonglong `ago, in a balaxy far far` away.
- `ago.*?far`: Longlonglong `ago, in a balaxy far` far away.

GreedinessExample.java

Boundary Matchers

- `^`: 一行的開頭
- `$`: 一行的結尾
- `\b`: 一行的開頭或結尾
- `\B`: 不是一行的開頭或結尾

Examples: it was the best of times or it was the worst of times

- `^it.*?times`: `it was the best of times` or it was the worst of times

- `\sit.*times$`: it was the best of times or `it was the worst of times`
- `\bor\b.{3}`: it was the best of times `or it` was the worst of times

BoundaryCharExample.java

java.util.regex.Matcher

- `replaceAll(String)`: 搜尋與取代所有內容
- `replaceFirst(String)`: 搜尋與取代第一個內容

Matching and Groups

- `simon.johnson@google.com`
- `(simon).(johnson)@(google.com)`
- `(1).(2)@(3)`
- `(\\S+?)\\. (\\S+?)@ (\\S+)`

MatchingExample.java