Bài 67: Tổng quan regular expression

- ✓ Giới thiệu
- ✓ Các siêu kí tự trong regular expression
- ✓ Lớp Pattern, lớp Matcher
- ✓ Minh họa

Giới thiệu

- Regular expression – biểu thức chính quy(chuối)
- Là một chuỗi kí tự được sắp xếp theo thứ tự đặc biệt dùng để so khớp, tìm kiếm chuỗi sử dụng cú pháp đặc trưng được thiết lập trong các pattern
- Chủ yếu được dùng để tìm kiếm, chỉnh sửa hoặc thao tác với dữ liệu dạng văn bản
- Liên quan đến regular expression có 3 lớp:
  - Lớp Pattern: đối tượng của lớp này là một đại diện đã biên dịch của biểu thức chính quy. Lớp Pattern không chứa public constructor do đó để tạo đối tượng của lớp ta dùng phương thức static compile(String regex)

Giới thiệu

- Liên quan đến regular expression có 3 lớp:
  - Lớp Matcher: đối tượng của lớp này thực hiện so khớp chuỗi đầu vào với mẫu so khớp. Lớp Matcher không có public constructor nên ta tạo đối tượng của nó qua phương thức matcher(String input) từ đối tượng của lớp Pattern
  - Lớp PatternSyntaxException: Lớp ngoại lệ của biểu thức chính quy chỉ ra ngoại lệ đã gặp phải. Đây là một runtime exception

Các siêu kí tự

Kí tự	Ý nghĩa
^	So khớp đầu dòng
$	So khớp cuối dòng
.	Khớp với bất kì kí tự nào ngoại trừ khoảng trắng
[abc]	Khớp với bất kì kí tự nào trong ngoặc vuông
[^abc]	Khớp với bất kì kí tự nào không nằm trong ngoặc vuông sau dấu ^
abc*	Khớp với 0 hoặc nhiều biểu thức abc nào đó
abc+	Khớp với 1 hoặc nhiều biểu thức abc nào đó
abc?	Khớp xem biểu thức abc có xuất hiện hay không
abc{ n }	Khớp với chính xác n lần xuất hiện của giá trị abc
abc{ n, }	Khớp với ít nhất n lần xuất hiện của biểu thức abc
abc{ m, n}	Khớp với ít nhất m, nhiều nhất n lần xuất hiện của biểu thức abc
a | b	Khớp với a hoặc b
(abc)	Gom nhóm biểu thức abc lại
\w	Khớp với các kí tự chữ cái hoặc chữ số
\W	Khớp với bất kì kí tự nào không phải chữ cái hoặc chữ số
\s	Khớp với khoảng trắng(\t, \n, \r, \f)
\S	Khớp với các kí tự không phải khoảng trắng
\d	Khớp với kí tự số (0-9)
\D	Khớp với kí tự không phải số
\A	Khớp với vị trí bắt đầu chuỗi kí tự
\Z	Khớp với vị trí cuối chuỗi kí tự, nếu cuối chuỗi là kí tự \n thì nó khớp với vị trí trước \n
\z	Khớp với vị trí cuối chuỗi kí tự
\G	Khớp với điểm mà lần so khớp trước đó kết thúc

Lớp Pattern

Một số phương thức thường dùng của lớp Pattern:

Phương thức	Mô tả
static Pattern compile(String regex)	Sử dụng để compile một biểu thức chính quy thành một mẫu so khớp thường dùng
static Pattern compile(String regex, int flags)	Sử dụng để compile một biểu thức chính quy thành một mẫu so khớp thường dùng với một cờ cho trước
int flags()	Lấy flag đang được sử dụng trong pattern hiện thời
Matcher matcher(CharSequence input)	Được sử dụng để tạo ra đối tượng matcher đem so khớp mẫu pattern đã có với chuỗi input vừa nhận được
Static boolean matches(String regex, CharSequence input)	Sử dụng để biên dịch biểu thức chính quy đã cho và thực hiện so khớp với input nhận được
String[] split(CharSequence input)	Sử dụng để tách chuỗi đầu vào tại các vị trí có mẫu so khớp. Trả về mảng các chuỗi con sau khi tách.

Lớp Matcher

Một số phương thức thường dùng của lớp Matcher:

Phương thức	Mô tả
int start()	Trả về vị trí bắt đầu khớp với mẫu pattern trong chuỗi input của lần so khớp đang xét
int end()	Trả về vị trí kế tiếp sau vị trí khớp với mẫu so khớp trong chuỗi đầu vào đang xét
boolean find()	Chủ yếu dùng để tìm sự xuất hiện nhiều lần của pattern trong chuỗi gốc
boolean find(int start)	Dùng để tìm sự xuất hiện của pattern trong chuỗi gốc từ vị trí start cho trước
int groupCount()	Sử dụng để trả về số nhóm trong pattern
boolean matches()	Dùng để test xem liệu pattern có khớp với input không
String replaceAll(String replacement)	dùng để thay thế toàn bộ chuỗi con khớp với pattern trong input bởi tham số được cung cấp. Trả về String kết quả
String replaceFirst(String replacement)	Thay thế chuỗi con đầu tiên trong chuỗi input khớp với pattern bởi tham số cho bởi phương thức. Trả về String kết quả

Ví dụ

➢Ví dụ thao tác với regex:

```
String regex = "\\s+"; // vị trí có 1 hoặc nhiều dấu cách
String str = "Hi there! I love learning Java programming " +
        "language. I hope you enjoy it too.";
Pattern pattern = Pattern.compile(regex);
Matcher matcher = pattern.matcher(str);

// kiểm tra xem mẫu pattern có khớp với chuỗi con nào trong
// str hay không
System.out.println("Chuỗi str chỉ chứa dấu cách? "
        + matcher.matches());

// tìm vị trí bắt đầu, kết thúc của regex trong str
while (matcher.find()) {
    System.out.println("Bắt đầu: " + matcher.start()
            + " - kết thúc: " + (matcher.end()));
}
```

Minh họa

- Thực hiện trong công cụ lập trình

Nội dung tiếp theo

Tạo regular expression kiểm tra định dạng mã sinh viên