**정규식 (Regular Expression)**

2016-11-24

이승진

**학습목표**

정규식 문법을 이해한다.

정규식 특수문자의 의미를 이해한다.

String 클래스의 정규식 메소드의 사용법을 학습한다.

**목차**

[1. 정규식 (regular expression) 2](#_Toc468357418)

[1) 정규식 특수 문자와 일반 문자 2](#_Toc468357419)

[2) [ ] 특수 문자 2](#_Toc468357420)

[3) . 특수 문자 3](#_Toc468357421)

[4) [^ ] 특수 문자 3](#_Toc468357422)

[5) ? 특수 문자 3](#_Toc468357423)

[6) \* 특수 문자 3](#_Toc468357424)

[7) + 특수 문자 4](#_Toc468357425)

[8) ^ 특수 문자 4](#_Toc468357426)

[9) $ 특수 문자 4](#_Toc468357427)

[10) {n} 특수 문자 4](#_Toc468357428)

[11) {n,m} 특수 문자 4](#_Toc468357429)

[12) ( ) 특수 문자 5](#_Toc468357430)

[13) | 특수 문자 5](#_Toc468357431)

[14) \ 특수 문자 5](#_Toc468357432)

[15) 연습문제 6](#_Toc468357433)

[16) \ 특수 문자와 Java 문자열 6](#_Toc468357434)

[17) \t \n 특수 문자 6](#_Toc468357435)

[2. String 클래스의 정규식 메소드 7](#_Toc468357436)

[1) String 클래스의 match 메소드 7](#_Toc468357437)

[2) String 클래스의 replaceAll, replaceFirst 메소드 8](#_Toc468357438)

[3) String 클래스의 split 메소드 11](#_Toc468357439)

[3. Pattern, Matcher 클래스 12](#_Toc468357440)

[1) 개요 12](#_Toc468357441)

[2) 정규식과 일치하는 부분 찾기 12](#_Toc468357442)

[3) 계속 찾기 13](#_Toc468357443)

[4) 정규식의 일부와 일치하는 문자열 14](#_Toc468357444)

[4. 실습문제 19](#_Toc468357445)

[5. 실습문제 답 23](#_Toc468357446)

[1) 제목 찾기 23](#_Toc468357447)

[2) idx 값과 제목 찾기 24](#_Toc468357448)

# 정규식 (regular expression)

텍스트 에디터에서 문자열을 찾기 & 바꾸기 기능을 생각해 보자. (find & replace 기능)

Windodws의 메모장의 찾기 & 바꾸기 기능은, 찾을 문자열과 완전히 동일한 문자열만 찾을 수 있다.

예를 들어 찾을 문자열이 **the** 이면 정확히 **the** 만 찾아지고 **The** 는 찾아지지 않는다.

문자열을 찾을 때, 완전히 동일한 문자열만 찾는 것이 아니고, 어떤 패턴과 일치하는 문자열을 전부 다 찾을 수 있으면 편리할 것이다.

찾을 문자열의 패턴을 지정하는 표현식(expression)이 정규식(regular expression) 이다.

예를 들어서 **the** 뿐만 아니라 **The** 도 찾을 수 있기 위한 정규식은 **[tT]he** 이다.

eclipse, visual studio 등 대부분의 개발 도구는 정규식 찾기 기능을 지원한다.

## 정규식 특수 문자와 일반 문자

정규식에서 특수 문자가 아닌 평범한 문자는, 정확히 일치하는 것만 찾는다.

예를 들어 정규식 **[tT]he** 에서 **he** 부분은 평범한 문자이다. 그래서 he 부분은 정확히 일치해야 한다.

정규식에서 특수 문자는 찾을 패턴을 지정한다.

## [ ] 특수 문자

괄호 안의 문자들 중에서 아무 것이나 한 문자와 일치하는지 비교한다

예1) **[tT]he**

**[tT]** 부분은 **t** 문자와도 일치하고 **T** 문자와도 일치한다.

**th** 부분은 특수 문자가 아니기 때문에 정확히 일치해야 한다.

그래서 **[tT]he** 정규식은 **the** 와도 일치하고 **The** 와도 일치한다.

예2) **9[123][0123]**

**9** 부분은 특수 문자가 아니기 때문에 정확히 일치해야 한다.

일치하는 문자열들: 910 911 912 913 920 921 922 923 930 931 932 933

일치하지 않는 문자열들: 900 901 902 903 914 915 916 833

예3) **[123][0-3]8**

**8** 부분은 특수 문자가 아니기 때문에 정확히 일치해야 한다.

**[0-3]** 부분은 **[0123]** 과 동일하다.

따라서 일치하는 문자열들도 **[123][0123]8** 정규식과 동일하다.

예4) **[1-9][0-9]a**

**a** 부분은 특수 문자가 아니기 때문에 정확히 일치해야 한다.

**[1-9][0-9]** 정규식은 **[123456789][0123456789]** 정규식과 동일하다.

일치하는 문자열들: 12a 23a 35a 78a 90a 23a 30a 98a 81a 74a 99a

일치하지 않는 문자열들: 00a 01a 02a 03a 04a 05a 06a 07a 08a 09a 99b

예5) **0x[0-9A-F][0-9A-F]**

**0x[0-9A-F][0-9A-F]** 정규식은 **0x[0123456789ABCDEF][0123456789ABCDEF]** 정규식과 동일하다.

**0x** 부분은 특수 문자가 아니기 때문에 정확히 일치해야 한다.

일치하는 문자열들: 0x2F 0x34 0xAB 0xEE

일치하지 않는 문자열들: x2F 0x3f

## . 특수 문자

아무 문자 한 개와 일치한다.

예1) **A.B**

일치하는 문자열: AaB AzB A3B A@B A#B

일치하지 않는 문자열: AB AacB AxyzB

예1) **A..B**

일치하는 문자열: AabB AxzB A3aB A4@B A!#B

일치하지 않는 문자열: AB AaB AxyzB

## [^ ] 특수 문자

괄호 안의 문자들을 제외한 다른 문자들 중에서 아무 것이나 한 문자와 일치하는지 비교한다

예1) **[^tT]he**

일치하는 문자열: ahe bhe che Ahe Bhe Che 3he #he @he

일치하지 않는 문자열: the The

예2) **[^0-9][A-F]**

**[^0-5][A-F]** 정규식은 **[^012345][ABCDEF]** 정규식과 동일하다.

일치하는 문자열: 6A 7B 8A 9F AC aB bF

일치하지 않는 문자열: 0A 1B 2C 5F af bH 6Z

## ? 특수 문자

? 특수 문자 바로 앞의 정규식과 일치하는 문자 0 ~ 1 개 있는지 비교한다.

예1) **AB?C**

A C 부분은 정확히 일치해야 한다.

B? 부분은 B 문자가 0 ~ 1 개만 있는지 비교한다.

일치하는 문자열: AC ABC

일치하지 않는 문자열: ABBC ADC BBC

예2) **A[d-f]?C**

A C 부분은 정확히 일치해야 한다.

[d-f]? 부분은 [d-f] 정규식과 일치하는 문자가 0 ~ 1 개만 있는지 비교한다.

일치하는 문자열: AC AdC AeC AfC

일치하지 않는 문자열: AddC AEC

## \* 특수 문자

\* 특수 문자 바로 앞의 정규식과 일치하는 문자 0 ~ 여러 개 있는지 비교한다.

예1) **AB\*C**

A C 부분은 정확히 일치해야 한다.

B\* 부분은 B 문자가 0 ~ 여러 개 있는지 비교한다.

일치하는 문자열: AC ABC ABBC ABBBC ABBBBC

일치하지 않는 문자열: ACC ADC BBC

예2) **A[d-f]\*C**

A C 부분은 정확히 일치해야 한다.

[d-f]\* 부분은 [d-f] 정규식과 일치하는 문자가 0 ~ 여러 개 있는지 비교한다.

일치하는 문자열: AC AdC AfeC AdeeC

일치하지 않는 문자열: AdgC AbeC AEC

## + 특수 문자

+ 특수 문자 바로 앞의 정규식과 일치하는 문자 1 ~ 여러 개 있는지 비교한다.

예1) AB+C

A C 부분은 정확히 일치해야 한다.

B+ 부분은 B 문자가 1 ~ 여러 개 있는지 비교한다.

일치하는 문자열: ABC ABBC ABBBC ABBBBC

일치하지 않는 문자열: AC CC ADC BBC

예2) A[d-f]+C

A C 부분은 정확히 일치해야 한다.

[d-f]+ 부분은 [d-f] 정규식과 일치하는 문자가 1 ~ 여러 개 있는지 비교한다.

일치하는 문자열: AdC AeC AfC AdefC AeeffC

일치하지 않는 문자열: AC AEC ADEC A3378C

## ^ 특수 문자

^ 특수 문자는 줄바꿈 문자로 구분되는 각 줄의 선두 부분에서 있는 문자열인지 검사한다.

예1) ^123

줄의 선두 부분에 있는 123 문자열과 일치한다.

줄의 선두가 아니면 일치하지 않는다.

## $ 특수 문자

$ 특수 문자는 줄바꿈 문자로 구분되는 각 줄의 끝 부분에서 있는 문자열인지 검사한다.

예1) abc$

줄의 끝 부분에 있는 abc 문자열과 일치한다.

줄의 끝 부분이 아니면 일치하지 않는다.

## {*n*} 특수 문자

{ } 괄호 안에는 숫자가 1 개 들어 있어야 한다.

{ } 괄호 바로 앞의 정규식과 일치하는 문자가 반복되는 수가, { } 괄호 안의 숫자와 일치해야 한다.

예1) AB{3}C

일치하는 문자열: ABBBC

예2) A[0-3]{4}C

일치하는 문자열: A0000C A0123C A0023C A1320C A3211C

## {*n,m*} 특수 문자

{ } 괄호 안에는 숫자가 2 개 들어 있어야 한다.

{ } 괄호 바로 앞의 정규식과 일치하는 문자가 반복되는 수가, n 보다 크거나 같고 m 보다 작거나 같아야 한다.

예1) AB{2,3}C

일치하는 문자열: ABBC ABBBC

예2) A[0-3]{2,4}C

일치하는 문자열: A00C A013C A0023C A3211C

## ( ) 특수 문자

( ) 괄호 안의 정규식을 묶는 역할을 한다.

예1) A(BC)?D

일치하는 문자열: AD ABCD

예2) A(BC){2,3}D

일치하는 문자열: ABCBCD ABCBCBCD

예3) A([a-c][0-3]){2}D

일치하는 문자열 Aa0b2D Ac2b0D Ab3c2D

일치하지 않는 문자열 Aba20D

예4) A(23)+D

일치하는 문자열: A23D A2323D A232323D

## | 특수 문자

| 특수 문자 앞 뒤의 정규식 둘 중의 하나와 문자열이 일치해야 한다.

예1) Ab|cD

| 특수 문자 앞의 정규식은 b 이고 뒤의 정규식은 c 이다.

위 정규식은 A[bc]D 정규식과 동일하다.

일치하는 문자열: AbD AcD

예2) A(bc)|(def)G

| 특수 문자 앞의 정규식은 ac 이고 뒤의 정규식은 def 이다.

일치하는 문자열: AbcG AdefG

## \ 특수 문자

정규식 특수 문자를 보통의 문자처럼 비교할 때, 특수 문자 앞에 \ 문자를 붙인다.

예1) Ab\?D

일치하는 문자열: Ab?D

예2) A[0-3]\+D

일치하는 문자열: A0+D A1+D A2+D A3+D

예3) \([1-9][0-9]\*\)

일치하는 문자열: (1) (2) (3) (10) (21) (34) (100) (235)

일치하지 않는 문자열: (0) (01) [10]

예4) \[1-9\]

일치하는 문자열: [1-9]

예5) [\t ]

탭 문자, 공백 문자 중의 한 문자와 일치한다.

예6) [\t ]+

탭 문자나 공백 문자가 1 개 이상 반복된 부분과 일치한다.

공백 문자 사이 사이에 탭 문자가 있어도 일치하고

탭 문자만 여러 개 반복되어도 일치하고

탭 문자 한 개나 공백 문자 한 개와도 일치한다.

## 연습문제

다음 정규식과 일치하는 문자열의 예를 적어라.

[a-zA-Z\_][a-zA-Z0-9\_]\*

[0-9]{6}-[1234][0-9]{6}

01[0-9]-[0-9]{3,4}-[0-9]{4}

\([1-9][0-9]\*\)

0x[0-9A-F]{2}

"%[0-9]\*[dsfx]"

[a-zA-Z]+@[a-zA-Z]+\.com

## \ 특수 문자와 Java 문자열

\ 문자는 Java 문자열에서도 특수 문자이다.

그래서 Java 문자열에서 \ 문자는 \ 문자로 해석되는 것이 아니고, \ 문자 다음 문자와 결합되어 해석된다.

예) "\n" Java 문자열은 \ 문자와 n 문자로 해석되지 않고, 줄바꿈 문자로 해석된다.

Java 문자열에서 \ 문자는 \\ 으로 입력해야 한다.

예를 들어 **Ab\?D** 정규식을 Java 문자열에 입력하려면 **"Ab\\?D"** 으로 입력해야 한다.

**\([1-9][0-9]\*\)** 정규식을 Java 문자열에 입력하려면 **"\\([1-9][0-9]\*\\)"** 으로 입력해야 한다.

## \t \n 특수 문자

정규식에서도 \t \n 문자는 각각 탭 문자와 줄바꿈 문자로 해석된다.

# String 클래스의 정규식 메소드

## String 클래스의 match 메소드

boolean match(String regex)

this 문자열이 정규식 regex에 일치하는지 검사하여 true/false를 리턴한다.

### Example1.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | package net.skhu.regex;  public class Example1 {    public static void main(String[] args) {  String[] a = new String[] { "010-323-1232", "02-542-3322", "010-9930-1123", "032-431-2323" };  String regex = "0(2|[1-9]{2})-[0-9]{3}-[0-9]{4}";  for (String s : a) {  if (s.matches(regex))  System.out.printf("\"%s\" matches \"%s\"\n", s, regex);  else  System.out.printf("\"%s\" doesn't match \"%s\"\n", s, regex);  }  }  } |

실행 결과 출력

|  |
| --- |
| "010-323-1232" doesn't match "0(2|[1-9]{2})-[0-9]{3}-[0-9]{4}"  "02-542-3322" matches "0(2|[1-9]{2})-[0-9]{3}-[0-9]{4}"  "010-9930-1123" doesn't match "0(2|[1-9]{2})-[0-9]{3}-[0-9]{4}"  "032-431-2323" matches "0(2|[1-9]{2})-[0-9]{3}-[0-9]{4}" |

### Example2.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | package net.skhu.regex;  public class Example2 {    public static void main(String[] args) {  String[] a = new String[] { "010-323-1232", "02-542-3322", "010-9930-1123", "032-431-2323" };  String regex = "010-[0-9]{3,4}-[0-9]{4}";  for (String s : a) {  if (s.matches(regex))  System.out.printf("\"%s\" matches \"%s\"\n", s, regex);  else  System.out.printf("\"%s\" doesn't match \"%s\"\n", s, regex);  }  }  } |

실행 결과 출력

|  |
| --- |
| "010-323-1232" matches "010-[0-9]{3,4}-[0-9]{4}"  "02-542-3322" doesn't match "010-[0-9]{3,4}-[0-9]{4}"  "010-9930-1123" matches "010-[0-9]{3,4}-[0-9]{4}"  "032-431-2323" doesn't match "010-[0-9]{3,4}-[0-9]{4}" |

## String 클래스의 replaceAll, replaceFirst 메소드

String replaceAll(String regex, String replacement)

this 문자열에서 정규식 regex와 일치하는 부분을 찾아서 replacement로 치환한 새 문자열을 리턴한다.

정규식과 일치하는 부분들을 전부 replacement로 치환한다.

새 문자열을 생성해서 리턴할 뿐 this 문자열은 변경되지 않는다.

String replaceFirst(String regex, String replacement)

this 문자열에서 정규식 regex와 일치하는 부분을 찾아서 replacement로 치환한 새 문자열을 리턴한다.

정규식과 일치하는 첫 부분만 replacement로 치환한다.

새 문자열을 생성해서 리턴할 뿐 this 문자열은 변경되지 않는다.

### Example3.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | package net.skhu.regex;  public class Example3 {    public static void main(String[] args) {  String[] a = new String[] {  "http://localhost:8080/test/list.jsp?pg=3&sz=15",  "http://localhost:8080/test/list.jsp?pg=9&sz=10",  "http://localhost:8080/test/list.jsp?pg=25&sz=20"  };  String regex = "pg=[0-9]+";  for (String s : a) {  String temp = s.replaceAll(regex, "pg=1");  System.out.printf("%s => %s\n", s, temp);  }  }  } |

실행 결과 출력

|  |
| --- |
| http://localhost:8080/test/list.jsp?pg=3&sz=15 => http://localhost:8080/test/list.jsp?pg=1&sz=15  http://localhost:8080/test/list.jsp?pg=9&sz=10 => http://localhost:8080/test/list.jsp?pg=1&sz=10  http://localhost:8080/test/list.jsp?pg=25&sz=20 => http://localhost:8080/test/list.jsp?pg=1&sz=20 |

### Example4.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | package net.skhu.regex;  public class Example4 {    public static void main(String[] args) {  String[] a = new String[] {  "http://localhost:8080/test/list.jsp?pg=3&sz=15",  "http://localhost:8080/test/list.jsp?pg=9&sz=10",  "http://localhost:8080/test/list.jsp?pg=25&sz=20"  };  String regex = "(\\?|&)[a-z]+=[0-9]+";  for (String s : a) {  String temp = s.replaceAll(regex, "");  System.out.printf("%s => %s\n", s, temp);  }  }  } |

실행 결과 출력

|  |
| --- |
| http://localhost:8080/test/list.jsp?pg=3&sz=15 => http://localhost:8080/test/list.jsp  http://localhost:8080/test/list.jsp?pg=9&sz=10 => http://localhost:8080/test/list.jsp  http://localhost:8080/test/list.jsp?pg=25&sz=20 => http://localhost:8080/test/list.jsp |

### Example5.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | package net.skhu.regex;  public class Example5 {    public static void main(String[] args) {  String[] a = new String[] {  "http://localhost:8080/test/list.jsp?pg=3&sz=15",  "http://localhost:8080/test/list.jsp?pg=9&sz=10",  "http://localhost:8080/test/list.jsp?pg=25&sz=20"  };  String regex = "(\\?|&)[a-z]+=[0-9]+";  for (String s : a) {  String temp = s.replaceFirst(regex, "");  System.out.printf("%s => %s\n", s, temp);  }  }  } |

실행 결과 출력

|  |
| --- |
| http://localhost:8080/test/list.jsp?pg=3&sz=15 => http://localhost:8080/test/list.jsp&sz=15  http://localhost:8080/test/list.jsp?pg=9&sz=10 => http://localhost:8080/test/list.jsp&sz=10  http://localhost:8080/test/list.jsp?pg=25&sz=20 => http://localhost:8080/test/list.jsp&sz=20 |

"(\\?|&)[a-z]+=[0-9]+"

Java에서 정규식 \ 특수문자를 \\ 으로 입력해야 함에 주의하자.

### Example6.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | package net.skhu.regex;  public class Example6 {    public static void main(String[] args) {  String s = "<table>\n" +  "\t<tr>\n\t\t <td>홍길동</td>\n\t\t <td>18</td>\n\t</tr>\n" +  "\t<tr>\n\t\t <td>전우치</td>\n\t\t <td>20</td>\n\t</tr>\n" +  "</table>\n";  System.out.printf("-------- (0)\n%s\n\n", s);    s = s.replaceAll("[ \t\n]+", "");  System.out.printf("-------- (1)\n%s\n\n", s);  s = s.replaceAll("</tr>", "\n");  System.out.printf("-------- (2)\n%s\n\n", s);  s = s.replaceAll("</td><td>", ", ");  System.out.printf("-------- (3)\n%s\n\n", s);  s = s.replaceAll("</?[a-zA-Z]+>", "");  System.out.printf("-------- (4)\n%s\n\n", s);  }  } |

실행 결과 출력

|  |
| --- |
| -------- (0)  <table>  <tr>  <td>홍길동</td>  <td>18</td>  </tr>  <tr>  <td>전우치</td>  <td>20</td>  </tr>  </table>  -------- (1)  <table><tr><td>홍길동</td><td>18</td></tr><tr><td>전우치</td><td>20</td></tr></table>  -------- (2)  <table><tr><td>홍길동</td><td>18</td>  <tr><td>전우치</td><td>20</td>  </table>  -------- (3)  <table><tr><td>홍길동, 18</td>  <tr><td>전우치, 20</td>  </table>  -------- (4)  홍길동, 18  전우치, 20 |

## String 클래스의 split 메소드

String[] split(String regex)

this 문자열에서 regex 정규식과 일치하는 부분을 기준으로 문자열을 잘라서

문자열 배열을 만들어 리턴한다.

this 문자열은 변경되지 않는다.

### Example7.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | package net.skhu.regex;  import java.util.Arrays;  public class Example7 {    public static void main(String[] args) {  String s = "one two, three,\t four; \t\nfive..,six";  String[] a = s.split("[ ,.;\t\n]+");  System.out.println(Arrays.toString(a));  }  } |

실행 결과 출력

|  |
| --- |
| [one, two, three, four, five, six] |

# Pattern, Matcher 클래스

String 클래스에도 정규식 찾기 & 바꾸기 메소드가 있지만,

좀 더 정교하게 정규식 찾기 & 바꾸기를 수행하기 위한 클래스들이 Pattern, Matcher 클래스이다.

## 개요

### Pattern 객체 생성

Pattern pattern = Pattern.compile("**정규식**");

**정규식**을 찾기 위한 Pattern 객체를 생성한다.

### Matcher 객체 생성

Matcher matcher = pattern.matcher("**비교할 문자열**");

**비교할 문자열**에서 **정규식**과 일치하는 부분을 찾기 위한 Matcher 객체를 생성한다.

### 찾기

matcher.find()

**비교할 문자열**에서 **정규식**과 일치하는 부분을 찾는다. 찾았으면 true를, 못찾았으면 false를 리턴한다.

### String의 matches 메소드와 Matcher의 find 메소드의 차이점

String 클래스의 matches 메소드는 입력 문자열 전체가 정규식과 일치해야 true를 리턴한다.

Matcher 클래스의 find 메소드는 정규식과 일치하는 부분이 입력 문자열에 포함되어 있으면 true를 리턴한다.

### 찾은 결과

matcher.group(0)

**비교할 문자열**에서 **정규식**과 일치하는 부분을 리턴한다.

## 정규식과 일치하는 부분 찾기

### Example1a.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | package net.skhu.regex2;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  public class Example1a {    public static void main(String[] args) {  String s = "Telephone: 032-431-2323 010-5533-2299 02-555-3388 010-222-5678";  String regex = "010-[0-9]{3,4}-[0-9]{4}";    Pattern pattern = Pattern.compile(regex);  Matcher matcher = pattern.matcher(s);  if (matcher.find())  System.out.println(matcher.group(0));  }  } |

실행결과 출력

|  |
| --- |
| 010-5533-2299 |

## 계속 찾기

### Example1b.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | package net.skhu.regex2;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  public class Example1b {    public static void main(String[] args) {  String s = "Telephone: 032-431-2323 010-5533-2299 02-555-3388 010-222-5678";  String regex = "010-[0-9]{3,4}-[0-9]{4}";    Pattern pattern = Pattern.compile(regex);  Matcher matcher = pattern.matcher(s);  while (matcher.find())  System.out.println(matcher.group(0));  }  } |

실행결과 출력

|  |
| --- |
| 010-5533-2299  010-222-5678 |

find 메소드를 반복해서 호출하면, 입력 문자열에서 정규식과 일치하는 부분을 계속 찾는다.

## 정규식의 일부와 일치하는 문자열

### Example1c.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | package net.skhu.regex2;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  public class Example1c {    public static void main(String[] args) {  String s = "Telephone: 032-431-2323 010-5533-2299 02-555-3388 010-222-5678";  String regex = "010-([0-9]{3,4})-([0-9]{4})";    Pattern pattern = Pattern.compile(regex);  Matcher matcher = pattern.matcher(s);  while (matcher.find()) {  System.out.printf("전체:%s ", matcher.group(0));  System.out.printf("국번:%s ", matcher.group(1));  System.out.printf("번호:%s\n", matcher.group(2));  }  }  } |

실행결과 출력

|  |
| --- |
| 전체:010-5533-2299 국번:5533 번호:2299  전체:010-222-5678 국번:222 번호:5678 |

### 정규식의 일부분

"010-[0-9]{3,4}-[0-9]{4}"

위 정규식와 아래의 정규식의 차이는 ( ) 괄호의 유무이다.

"010-([0-9]{3,4})-([0-9]{4})"

정규식에서 ( ) 괄호는 정규식의 일부분을 묶을 때 사용한다.

정규식에서 ( ) 괄호와 일치하는 부분만 알고 싶을 때 group(int 순서) 메소드를 사용한다.

Matcher의 group(0) 메소드는 정규식과 일치하는 부분 전체를 리턴한다.

Matcher의 group(1) 메소드는 정규식과 일치하는 부분 중에서 첫 번째 ( ) 괄호에 해당하는 부분을 리턴한다.

Matcher의 group(2) 메소드는 정규식과 일치하는 부분 중에서 두 번째 ( ) 괄호에 해당하는 부분을 리턴한다.

### Example1d.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | package net.skhu.regex2;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  public class Example1d {    public static void main(String[] args) {  String s = "Telephone: 032-431-2323 010-5533-2299 02-555-3388 010-222-5678";  String regex = "010-(?<a>[0-9]{3,4})-(?<b>[0-9]{4})";    Pattern pattern = Pattern.compile(regex);  Matcher matcher = pattern.matcher(s);  while (matcher.find()) {  System.out.printf("전체:%s ", matcher.group(0));  System.out.printf("국번:%s ", matcher.group("a"));  System.out.printf("번호:%s\n", matcher.group("b"));  }  }  } |

실행결과 출력

|  |
| --- |
| 전체:010-5533-2299 국번:5533 번호:2299  전체:010-222-5678 국번:222 번호:5678 |

### 정규식의 일부분

"010-([0-9]{3,4})-([0-9]{4})"

위 정규식와 아래의 정규식의 차이는 **?<이름>** 부분의 유무이다.

"010-(?<a>[0-9]{3,4})-(?<b>[0-9]{4})"

(**?<이름>**정규식)

위 정규식에서 **?<이름>** 부분은 괄호로 묶인 정규식에 이름을 부여한다.

이 괄호와 일치하는 부분만 알고 싶을 때 group(String 이름) 메소드를 사용한다.

### Example2a.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | package net.skhu.regex2;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  public class Example2a {    public static void main(String[] args) {  String[] a = new String[] {  "http://localhost:8080/test/list.jsp?pg=3&sz=15",  "http://localhost:8080/test/list.jsp?pg=9&sz=10",  "http://localhost:8080/test/list.jsp?pg=25&sz=20&od=1"  };  String regex = "(\\?|&)[a-z]+=[0-9]+";  Pattern pattern = Pattern.compile(regex);    for (String s : a) {  System.out.printf("\n%s ", s);  Matcher matcher = pattern.matcher(s);  while (matcher.find())  System.out.printf("%s ", matcher.group(0));  }  }  } |

실행결과 출력

|  |
| --- |
| http://localhost:8080/test/list.jsp?pg=3&sz=15 ?pg=3 &sz=15  http://localhost:8080/test/list.jsp?pg=9&sz=10 ?pg=9 &sz=10  http://localhost:8080/test/list.jsp?pg=25&sz=20&od=1 ?pg=25 &sz=20 &od=1 |

### Example2b.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | package net.skhu.regex2;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  public class Example2b {    public static void main(String[] args) {  String[] a = new String[] {  "http://localhost:8080/test/list.jsp?pg=3&sz=15",  "http://localhost:8080/test/list.jsp?pg=9&sz=10",  "http://localhost:8080/test/list.jsp?pg=25&sz=20&od=1"  };  String regex = "(\\?|&)([a-z]+)=([0-9]+)";  Pattern pattern = Pattern.compile(regex);    for (String s : a) {  System.out.printf("\n\n%s\n", s);  Matcher matcher = pattern.matcher(s);  while (matcher.find())  System.out.printf("group1:%s group2:%s group3:%s\n",  matcher.group(1), matcher.group(2), matcher.group(3));  }  }    } |

실행결과 출력

|  |
| --- |
| http://localhost:8080/test/list.jsp?pg=3&sz=15  group1:? group2:pg group3:3  group1:& group2:sz group3:15  http://localhost:8080/test/list.jsp?pg=9&sz=10  group1:? group2:pg group3:9  group1:& group2:sz group3:10  http://localhost:8080/test/list.jsp?pg=25&sz=20&od=1  group1:? group2:pg group3:25  group1:& group2:sz group3:20  group1:& group2:od group3:1 |

### Example2c.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | package net.skhu.regex2;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  public class Example2c {    public static void main(String[] args) {  String[] a = new String[] {  "http://localhost:8080/test/list.jsp?pg=3&sz=15",  "http://localhost:8080/test/list.jsp?pg=9&sz=10",  "http://localhost:8080/test/list.jsp?pg=25&sz=20&od=1"  };  String regex = "(\\?|&)(?<**name**>[a-z]+)=(?<**value**>[0-9]+)";  Pattern pattern = Pattern.compile(regex);    for (String s : a) {  System.out.printf("\n\n%s\n", s);  Matcher matcher = pattern.matcher(s);  while (matcher.find())  System.out.printf("name:%s value:%s\n", matcher.group("**name**"), matcher.group("**value**"));  }  }    } |
|  |  |

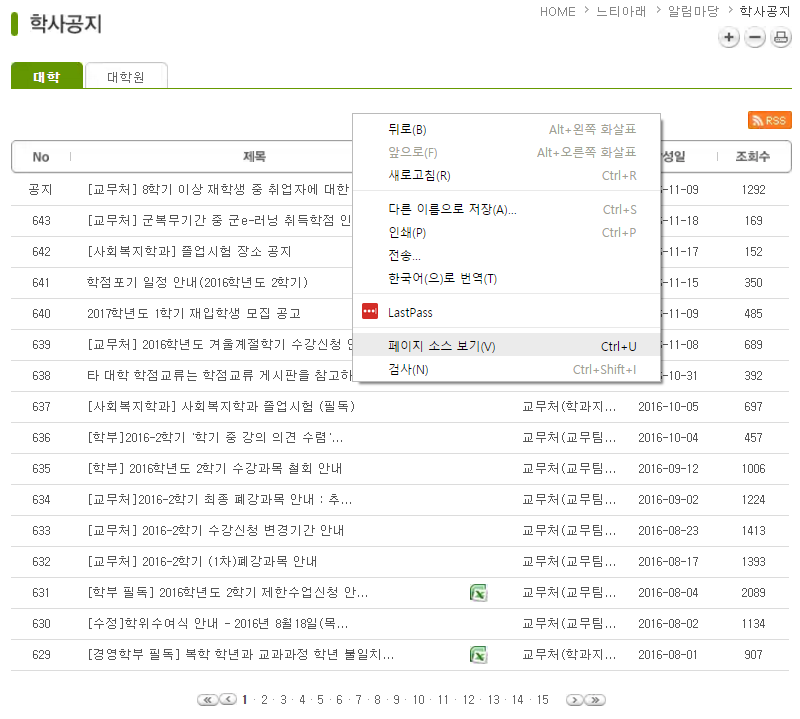
실행결과 출력

|  |
| --- |
| http://localhost:8080/test/list.jsp?pg=3&sz=15  name:pg value:3  name:sz value:15  http://localhost:8080/test/list.jsp?pg=9&sz=10  name:pg value:9  name:sz value:10  http://localhost:8080/test/list.jsp?pg=25&sz=20&od=1  name:pg value:25  name:sz value:20  name:od value:1 |

# 실습문제

성공회대 홈페이지 게시판

http://www.skhu.ac.kr/board/boardlist.aspx?bsid=10004&searchBun=51



"소스 보기" 메뉴를 클릭하면 다음과 같은 HTML 소스를 볼 수 있다.

|  |
| --- |
| **. . . 생략 . . .**  <table summary="NO, 제목, 첨부파일, 작성자, 작성일, 조회수 구분한 일반 게시판 목록 입니다." class="board\_list">  <caption>공지사항 게시판 목록</caption>  <colgroup>  <col width="55px" />    <col width="" />  <col width="75px" />  <col width="90px" />  <col width="90px" />  <col width="65px" />  </colgroup>  <thead>  <tr>  <th scope="col" abbr="NO"><img src="/common/images/txt/txt\_board\_list\_th\_no.gif" alt="NO" /></th>    <th scope="col" abbr="제목"><img src="/common/images/txt/txt\_board\_list\_th\_title.gif" alt="제목" /></th>  <th scope="col" abbr="첨부파일"><img src="/common/images/txt/txt\_board\_list\_th\_file.gif" alt="첨부파일" /></th>  <th scope="col" abbr="작성자"><img src="/common/images/txt/txt\_board\_list\_th\_write.gif" alt="작성자" /></th>  <th scope="col" abbr="작성일"><img src="/common/images/txt/txt\_board\_list\_th\_date.gif" alt="작성일" /></th>  <th scope="col" abbr="조회수" class="last"><img src="/common/images/txt/txt\_board\_list\_th\_inquiry.gif" alt="조회수" /></th>  </tr>  </thead>  <tbody>    <tr>  <td>공지</td>    <td class="left15"><span id="ctl00\_ContentPlaceHolder1\_ctl00\_rptList2\_ctl00\_lblTitle"><a href='boardread.aspx?idx=27461&amp;curpage=1&amp;bsid=10004&amp;searchBun=51'>[교무처] 8학기 이상 재학생 중 취업자에 대한 공...</a></span></td>  <td><a href='/common/Handler/BoardDownload.aspx?idx=27461&amp;bsid=10004&amp;threadsid=2743&amp;fsid=9097' title='파일 다운로드'><img src="/common/images/common/btn/btn\_file\_Adobe.gif" alt="pdf" /></a></td>  <td>교무처(교무팀...</td>  <td>2016-11-09</td>  <td>1292</td>  </tr>      <tr>  <td><span id="ctl00\_ContentPlaceHolder1\_ctl00\_rptList\_ctl00\_lblNum">643</span></td>    <td class="left15"><span id="ctl00\_ContentPlaceHolder1\_ctl00\_rptList\_ctl00\_lblTitle"><a href='boardread.aspx?idx=27536&amp;curpage=1&amp;bsid=10004&amp;searchBun=51'>[교무처] 군복무기간 중 군e-러닝 취득학점 인정안...</a></span></td>  <td><a href='/common/Handler/BoardDownload.aspx?idx=27536&amp;bsid=10004&amp;threadsid=2752&amp;fsid=9146' title='파일 다운로드'><img src="/common/images/common/btn/btn\_file\_hwp.gif" alt="한글" /></a></td>  <td>교무처(교무팀...</td>  <td>2016-11-18</td>  <td>169</td>  </tr>    <tr>  <td><span id="ctl00\_ContentPlaceHolder1\_ctl00\_rptList\_ctl01\_lblNum">642</span></td>    <td class="left15"><span id="ctl00\_ContentPlaceHolder1\_ctl00\_rptList\_ctl01\_lblTitle"><a href='boardread.aspx?idx=27516&amp;curpage=1&amp;bsid=10004&amp;searchBun=51'>[사회복지학과] 졸업시험 장소 공지</a></span></td>  <td></td>  <td>교무처(학과지...</td>  <td>2016-11-17</td>  <td>152</td>  </tr>    <tr>  <td><span id="ctl00\_ContentPlaceHolder1\_ctl00\_rptList\_ctl02\_lblNum">641</span></td>    <td class="left15"><span id="ctl00\_ContentPlaceHolder1\_ctl00\_rptList\_ctl02\_lblTitle"><a href='boardread.aspx?idx=27496&amp;curpage=1&amp;bsid=10004&amp;searchBun=51'>학점포기 일정 안내(2016학년도 2학기) </a></span></td>  <td></td>  <td>교무처(교무팀...</td>  <td>2016-11-15</td>  <td>350</td>  </tr> |

노란색 형광색으로 표시한 부분이 게시글 제목이다.

작업 절차

① http://www.skhu.ac.kr/board/boardlist.aspx?bsid=10004&searchBun=51

페이지의 HTML 소스를 **홈페이지\_게시판.html** 파일에 저장하여 강의자료 웹폴더에 업로드했음.

이 파일을 다운로드 하라.

② **홈페이지\_게시판.html** 텍스트 파일의 전체 내용을 읽어서 String 변수에 대입하라.

③ 정규식 찾기 기능을 활용하여 게시글 제목만 추출하라.

④ 게시글 제목 목록을 화면에 출력하라.

### 참고: 텍스트 파일 읽기

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | package net.skhu.file;  import java.io.BufferedReader;  import java.io.FileInputStream;  import java.io.IOException;  import java.io.InputStreamReader;  public class Example1 {    static String readTextFile(String path) throws IOException {  try(BufferedReader reader = new BufferedReader(  new InputStreamReader(new FileInputStream(path), "EUC-KR"))) {  StringBuilder builder = new StringBuilder();  String s = reader.readLine();  while (s != null) {  builder.append(s);  builder.append("\n");  s = reader.readLine();  }  return builder.toString();  }  }    public static void main(String[] args) throws IOException {  String path = "d:/temp/홈페이지\_게시판.html";  String s = readTextFile(path);  System.out.println(s);  }  } |

위 예제에서 수정할 부분은 텍스트 파일 경로명과, 텍스트 파일의 인코딩이다.

홈페이지\_게시판.html 텍스트 파일의 인코딩은 "EUC-KR" 이다.

만약 읽어야할 텍스트 파일의 인코딩이 "UTF-8" 이면, InputStreamReader 생성자의 두 번째 파라미터를 그렇게 수정해 주어야 한다.

### 좀 더 간결한 구현 방법

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | package net.skhu.file2;  import java.io.IOException;  import java.nio.charset.Charset;  import java.nio.file.Files;  import java.nio.file.Paths;  public class Example1 {  static String readTextFile(String path, Charset encoding) throws IOException {  byte[] encoded = Files.readAllBytes(Paths.get(path));  return new String(encoded, encoding);  }  public static void main(String[] args) throws IOException {  String path = "d:/temp/홈페이지\_게시판.html";  String s = readTextFile(path, Charset.forName("EUC-KR"));  System.out.println(s);  }  } |

### 힌트

<a href='boardread.aspx.+>

위 패턴으로 찾을 때의 문제점은 아래와 같다.

<a href='boardread.aspx?idx=27461&amp;curpage=1&amp;bsid=10004&amp;searchBun=51'>a</a>

위와 같이 찾을 수도 있고

<a href='boardread.aspx?idx=27599&amp;curpage=1&amp;bsid=10004&amp;searchBun=51'>b</a>

위와 같이 찾을 수도 있다.

<a href='boardread.aspx[^>]+>

위 패턴으로 찾으면

<a href='boardread.aspx?idx=27590&amp;curpage=1&amp;bsid=10004&amp;searchBun=51'>c</a>

위와 같이 찾을 수만 있다.

제목을 찾기 위한 정규식

<a href='boardread.aspx[^>]+>([^<]+)</a>

<a href='boardread.aspx?idx=27590&amp;curpage=1&amp;bsid=10004&amp;searchBun=51'>c</a>

제목과 idx 값을 찾기 위한 정규식

<a href='boardread.aspx\\?idx=([0-9]+)[^>]+>([^<]+)</a>

<a href='boardread.aspx?idx=27590&amp;curpage=1&amp;bsid=10004&amp;searchBun=51'>c</a>

# 실습문제 답

## 제목 찾기

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | package net.skhu.file2;  import java.io.IOException;  import java.nio.charset.Charset;  import java.nio.file.Files;  import java.nio.file.Paths;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  public class Example2 {  static String readTextFile(String path, Charset encoding) throws IOException {  byte[] encoded = Files.readAllBytes(Paths.get(path));  return new String(encoded, encoding);  }  public static void main(String[] args) throws IOException {  String path = "d:/temp/홈페이지\_게시판.html";  String s = readTextFile(path, Charset.forName("EUC-KR"));  String regex = "<a href='boardread.aspx[^>]+>([^<]+)</a>";  Pattern pattern = Pattern.compile(regex);  Matcher matcher = pattern.matcher(s);  while (matcher.find())  System.out.println(matcher.group(1));  }  } |

출력

|  |
| --- |
| [교무처] 8학기 이상 재학생 중 취업자에 대한 공...  [교무처] 군복무기간 중 군e-러닝 취득학점 인정안...  [사회복지학과] 졸업시험 장소 공지  학점포기 일정 안내(2016학년도 2학기)  2017학년도 1학기 재입학생 모집 공고  [교무처] 2016학년도 겨울계절학기 수강신청 안내...  타 대학 학점교류는 학점교류 게시판을 참고하세요  [사회복지학과] 사회복지학과 졸업시험 (필독)  [학부]2016-2학기 '학기 중 강의 의견 수렴'...  [학부] 2016학년도 2학기 수강과목 철회 안내  [교무처]2016-2학기 최종 폐강과목 안내 : 추...  [교무처] 2016-2학기 수강신청 변경기간 안내  [교무처] 2016-2학기 (1차)폐강과목 안내  [학부 필독] 2016학년도 2학기 제한수업신청 안...  [수정]학위수여식 안내 - 2016년 8월18일(목...  [경영학부 필독] 복학 학년과 교과과정 학년 불일치... |

## idx 값과 제목 찾기

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | package net.skhu.file2;  import java.io.IOException;  import java.nio.charset.Charset;  import java.nio.file.Files;  import java.nio.file.Paths;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  public class Example3 {  static String readTextFile(String path, Charset encoding) throws IOException {  byte[] encoded = Files.readAllBytes(Paths.get(path));  return new String(encoded, encoding);  }  public static void main(String[] args) throws IOException {  String path = "d:/temp/홈페이지\_게시판.html";  String s = readTextFile(path, Charset.forName("EUC-KR"));  String regex = "<a href='boardread.aspx\\?idx=([0-9]+)[^>]+>([^<]+)</a>";  Pattern pattern = Pattern.compile(regex);  Matcher matcher = pattern.matcher(s);  while (matcher.find())  System.out.printf("%s %s\n", matcher.group(1), matcher.group(2));  }  } |

실행결과 출력

|  |
| --- |
| 27461 [교무처] 8학기 이상 재학생 중 취업자에 대한 공...  27536 [교무처] 군복무기간 중 군e-러닝 취득학점 인정안...  27516 [사회복지학과] 졸업시험 장소 공지  27496 학점포기 일정 안내(2016학년도 2학기)  27462 2017학년도 1학기 재입학생 모집 공고  27457 [교무처] 2016학년도 겨울계절학기 수강신청 안내...  27404 타 대학 학점교류는 학점교류 게시판을 참고하세요  27228 [사회복지학과] 사회복지학과 졸업시험 (필독)  27218 [학부]2016-2학기 '학기 중 강의 의견 수렴'...  27066 [학부] 2016학년도 2학기 수강과목 철회 안내  26979 [교무처]2016-2학기 최종 폐강과목 안내 : 추...  26876 [교무처] 2016-2학기 수강신청 변경기간 안내  26844 [교무처] 2016-2학기 (1차)폐강과목 안내  26745 [학부 필독] 2016학년도 2학기 제한수업신청 안...  26717 [수정]학위수여식 안내 - 2016년 8월18일(목...  26700 [경영학부 필독] 복학 학년과 교과과정 학년 불일치... |