

11. 정규 표현식, Enum, Random



자바 JDK로 프로그래밍 날개 달기



정규 표현식(Regular Expression)

❖ 정규 표현식(Regular Expression)

- 특정한 규칙을 가진 문자열의 집합을 표현하고 처리하는 것을 말한다.
- 데이터의 유효성 검사에 주로 사용된다.
- 자주 사용하는 정규 표현식

표현식	설 명
^[0-9]*\$	숫자
^[a-zA-Z]*\$	영문 대, 소문자
^[가-힣]*\$	한글
^010[-](d{3} d{4})[-]d{4}\$	휴대폰
^d{6}[-][1-4]{6}\$	주민등록번호
^d{3}[-]d{2}\$	우편번호

메타문자	설 명
^	정규식 시작
\$	정규식 끝
*	문자 반복(+ 도 가능)
{3}	반복횟수
[x]	x를 찾음
\d	숫자
\s	공백
\w	알파벳+숫자



정규 표현식(Regular Expression)

❖ 정규 표현식(Regular Expression)

메타 문자	설명	예시 (<code>Pattern.compile()</code> 인자)
<code>*</code>	0번 이상 반복	<code>"a*b"</code> → <code>"b"</code> , <code>"ab"</code> , <code>"aaab"</code>
<code>+</code>	1번 이상 반복	<code>"a+b"</code> → <code>"ab"</code> , <code>"aaab"</code> (but not <code>"b"</code>)
<code>?</code>	0번 또는 1번	<code>"a?b"</code> → <code>"b"</code> , <code>"ab"</code>
<code>^</code>	문자열의 시작	<code>"^a"</code> → <code>"a"</code> 로 시작하는 문자열
<code>\$</code>	문자열의 끝	<code>"a\$"</code> → <code>"a"</code> 로 끝나는 문자열
<code>[0-9]</code>	0부터 9까지의 숫자	<code>"[0-9]+"</code> → <code>"123"</code>
<code>[a-z]</code>	소문자 a부터 z까지	<code>"[a-z]+"</code> → <code>"hello"</code>
<code>\d</code>	숫자 (<code>[0-9]</code> 와 동일)	<code>"\d+"</code> → <code>"123"</code>
<code>\w</code>	단어 문자 (알파벳, 숫자, <code>_</code>)	<code>"\w+"</code> → <code>"abc_123"</code>



정규 표현식(Regular Expression)

❖ 정규 표현식(Regular Expression)

Package java.util.regex

Class Pattern

java.lang.Object
java.util.regex.Pattern

All Implemented Interfaces:

Serializable

```
public final class Pattern
extends Object
implements Serializable
```

A compiled representation of a regular expression.

A regular expression, specified as a string, must first be compiled into a pattern object that can match arbitrary character sequences against the regular expression. Multiple matchers can share the same pattern.

A typical invocation sequence is thus

```
Pattern p = Pattern.compile("a*b");
Matcher m = p.matcher("aaaaab");
boolean b = m.matches();
```



정규 표현식(Regular Expression)

❖ Pattern, Matcher 클래스 활용

```
//a*b 패턴 검사 - a가 0번 이상 반복
//a+b 패턴 - a가 1번 이상 반복
Pattern pat = Pattern.compile("a*b");
Matcher m = pat.matcher("aaab"); //a가 없어도 true
boolean b1 = m.matches();

System.out.println(b1);

//숫자만 허용하는 패턴 검사
String pattern = "[0-9]*$";
String str = "abc1031";

boolean b2 = Pattern.matches(pattern, str);
System.out.println(b2);
```

```
true
false
```



정규 표현식(Regular Expression)

❖ Pattern, Matcher 클래스 활용

```
//한글과 전화번호 패턴 검사
String name = "제갈수현";
String tel = "010-1234-5678";

boolean name_check = Pattern.matches("[가-힣]{2,5}$", name);
boolean tel_check = Pattern.matches("^010[-](\\d{3}|\\d{4})[-]\\d{4}$", tel);

System.out.println("이름 검사: " + name_check);
System.out.println("전화번호 검사: " + tel_check);

//한글 이름 패턴 유효성 검사
Scanner sc = new Scanner(System.in);
System.out.print("한글 이름을 입력하세요: ");
String inputName = sc.nextLine();

if (!Pattern.matches("[가-힣]{2,5}$", inputName)) {
    System.out.println("올바른 한글 이름이 아닙니다!");
}
System.out.println("이름: " + inputName);

sc.close();
```

이름 검사: true
전화번호 검사: true
한글 이름을 입력하세요: 제갈수현
이름: 제갈수현



정규 표현식(Regular Expression)

❖ replaceAll() 메서드 활용

```
//비밀번호 보안 처리
String password = "P@ssw0rd!";
//^ - 부정 문자(아니다)
String masked = password.replaceAll("[^a-zA-Z0-9]", "*");

System.out.println(masked); // P*ssw0rd*

//게시글 금칙어 처리
String text = "안녕@하세요! #스팸";
// 한글과 공백만 허용
String filtered = text.replaceAll("[^ㄱ-힣\\s]", "*");

System.out.println(filtered); // 안녕*하세요* **
```



열거 타입(enum)

■ 열거 타입

한정된 값인 열거 상수 중에서 하나의 상수를 저장하는 타입이다.

```
public enum Season {  
    봄,  
    여름,  
    가을,  
    겨울  
}
```

```
Season season = null;
```

```
season = Season.여름
```



열거 타입(enum)

■ 열거 타입

```
public class SeasonTest {  
  
    public static void main(String[] args) {  
        Season season = null;  
        season = Season.여름;  
  
        switch(season) {  
            case 봄:  
                season = Season.봄;  
                break;  
            case 여름:  
                season = Season.여름;  
                break;  
            case 가을:  
                season = Season.가을;  
                break;  
            case 겨울:  
                season = Season.겨울;  
                break;  
        }  
        System.out.println("현재 계절은 " + season + "입니다.");  
  
        if(season == Season.여름) {  
            System.out.println("무더위와 장마가 옵니다.");  
        }else {  
            System.out.println("무더위와 장마가 별로 없습니다.");  
        }  
    }  
}
```



열거 타입(enum)

■ 열거 타입

```
enum Level{ //열거형 상수
    LOW,
    MEDIUM,
    HIGH
}

public class EnumLevel {
    public static void main(String[] args) {
        Level level = Level.HIGH; //상수이므로 new를 사용하지 않음

        switch(level) {
            case LOW:
                System.out.println("Low level");
                break;
            case MEDIUM:
                System.out.println("Medium level");
                break;
            case HIGH:
                System.out.println("High level");
                break;
            default:
                System.out.println("레벨이 없습니다.");
                break;
        }
    }
}
```



열거 타입(enum)

```
//1, 2, 3... 순서로 나열됨
enum Week{
    SUNDAY,
    MONDAY,
    TUESDAY,
    WEDNESDAY,
    THURSDAY,
    FRIDAY,
    SATURDAY
}
```

```
Week today = null; //enum 객체 생성
```

```
Calendar cal = Calendar.getInstance(); //Calendar 객체 생성
//요일 가져옴(1-일, 2-월, 3-화, 4-수, 5-목, 6-금, 7-토)
int week = cal.get(Calendar.DAY_OF_WEEK);
//System.out.println(week);
```

```
switch(week) {
case 1:
    today = Week.SUNDAY; break;
case 2:
    today = Week.MONDAY; break;
case 3:
    today = Week.TUESDAY; break;
case 4:
    today = Week.WEDNESDAY; break;
case 5:
    today = Week.THURSDAY; break;
case 6:
    today = Week.FRIDAY; break;
case 7:
    today = Week.SATURDAY; break;
```



열거 타입(enum)

```
default:
    System.out.println("요일이 없습니다."); break;
}
System.out.println("Today is " + today);

if(today == Week.SUNDAY) {
    System.out.println("일요일에는 놀러 나갑니다.");
}else {
    System.out.println("평일에는 열심히 코딩합니다.");
}
```

```
5
Today is THURSDAY
평일에는 열심히 코딩합니다.
```



난수 생성

● 난수 생성 비교

방법	특징	사용 예시
Math.random()	$0.0 \leq x < 1.0$ (Double)	<code>(int)(Math.random() * 6) + 1</code>
Random 클래스	다양한 자료형 지원	<code>rnd.nextInt(6) + 1</code>

● Random 클래스

난수(무작위수)를 생성하는 클래스이다.

Random

```
public Random(long seed)
```

Creates a new random number generator using a single

Implementation Requirements:

The invocation `new Random(seed)` is equivalent to:

```
Random rnd = new Random();  
rnd.setSeed(seed);
```

Parameters:

seed - the initial seed

See Also:

`setSeed(long)`



Random 클래스

● Random 클래스 활용

```
//난수 생성 - Math.random()
int n1 = (int)(Math.random() * 2); //0 ~ 1
System.out.println(n1);

//난수 생성 - Random() 클래스 활용
Random rnd = new Random();
//rnd.setSeed(0); //시드 설정 - 동일한 결과 반복

//시스템 시간을 기반으로 자동 변화
System.out.println(rnd.nextInt());
int n2 = rnd.nextInt(2);
System.out.println(n2); //0 ~ 1

//동전 던지기
int coin = rnd.nextInt(2) + 1;
if(coin == 1)
    System.out.println("앞면");
else
    System.out.println("뒷면");
```



Random 클래스

● Random 클래스를 활용한 주사위 던지기 게임

```
Random random = new Random();
int dice = random.nextInt(6) + 1;
//System.out.println(dice);

int dice1, dice2, total;
for(int i=0; i<10; i++) {
    dice1 = random.nextInt(6) + 1;
    dice2 = random.nextInt(6) + 1;
    total = dice1 + dice2;
    System.out.println(total);
    if(total==7)
        System.out.println("Seven Thrown!!");
    if(total==10)
        System.out.println("Ten Thrown!!");
    if(dice1==dice2)
        System.out.println("Double Thrown!!");
}
```

```
7
Seven Thrown!!
4
Double Thrown!!
7
Seven Thrown!!
6
4
Double Thrown!!
7
Seven Thrown!!
6
6
8
4
```



Random 클래스

- Lotto 복권 추첨

```
int[] lotto = new int[6]; //로또 6자리 공간

Random rand = new Random();
System.out.print("당첨 번호 : ");
for(int i=0; i<6; i++) {
    lotto[i] = rand.nextInt(45) + 1;
    System.out.print(lotto[i] + " ");
}
```

당첨 번호 : 44 27 4 15 41 7



Random 클래스

● Lotto 복권 추첨 - 당첨 여부 판정

```
// 선택 번호
int[] selNumber = new int[6]; // 선택 번호 6개가 저장될 배열 생성
int i;

Random rand = new Random(3); //선택 번호를 얻기 위한 객체 생성(seed값)
System.out.print("선택 번호 : ");
for(i=0; i<6; i++) {
    selNumber[i] = rand.nextInt(45) + 1;
    System.out.print(selNumber[i] + " ");
}
System.out.println();

// 당첨 번호
int[] winNumber = new int[6];
rand = new Random(3); //당첨 번호를 얻기 위해 객체 생성
System.out.print("당첨 번호: ");
for(i=0; i<6; i++) {
    winNumber[i] = rand.nextInt(45) + 1;
    System.out.print(winNumber[i] + " ");
}
System.out.println();
```



Lotto 복권 추첨

● Lotto 복권 추첨 - 당첨 여부 판정

```
// 당첨 여부
boolean result = Arrays.equals(selNumber, winNumber);
System.out.print("당첨 여부 : ");
if(result) {
    System.out.println("1등에 당첨되었습니다.");
}
else {
    System.out.println("당첨되지 않았습니다.");
}
```

선택 번호 : 15 21 16 17 34 28
당첨 번호: 15 21 16 17 34 28
당첨 여부 : 1등에 당첨되었습니다.

```
public static boolean equals(int[] a, int[] a2) {
    if (a==a2)
        return true;
    if (a==null || a2==null)
        return false;

    int length = a.length;
    if (a2.length != length)
        return false;

    return ArraysSupport.mismatch(a, a2, length) < 0;
}
```



숫자 추측 게임

● 숫자 추측 게임

숫자(1~30)를 입력하세요:20
너무 작아요!
숫자(1~30)를 입력하세요:25
너무 작아요!
숫자(1~30)를 입력하세요:27
정답!

```
Scanner sc = new Scanner(System.in);
Random rand = new Random();
int comNum = rand.nextInt(30) + 1; //1 ~ 30

//System.out.println(com);
while(true) {
    System.out.print("숫자(1~30)를 입력하세요:");
    int guessNum = sc.nextInt();
    if(guessNum == comNum) {
        System.out.println("정답!");
        break;
    }else if(guessNum > comNum) {
        System.out.println("너무 커요!");
    }else {
        System.out.println("너무 작아요!");
    }
}
sc.locale();
```



문자열 자르기 – split()

- String 클래스의 split() 사용

```
public class SplitTest {  
    public static void main(String[] args) {  
        //문자열 자르기  
        String str = "봄 여름 가을 겨울";  
        String[] season = str.split(" "); //배열 생성  
  
        Random rnd = new Random();  
        int rndIdx = rnd.nextInt(season.length);  
  
        System.out.println(season[rndIdx]);  
    }  
}
```



영어 타자 게임

● 영어 타자 게임

- 게임이 시작되면 영어 단어가 화면에 표시된다.
- 사용자는 최대한 빠르고 정확하게 입력해야 한다.
- 바르게 입력했으면 다음 문제로 넘어가고 "통과"를 출력한다.
- 오타가 있으면 같은 단어가 한 번 더 나온다.
- 타자 게임 시간을 측정한다.

영어타자 게임, 준비되면 엔터

문제1

moon

moon

통과!

문제2

tree

tree

통과!

문제3

moon

오타! 다시 도전!

문제9

cow

cow

통과!

문제10

mountain

mountain

통과!

게임 소요 시간35초입니다.



영어 타자 게임

● 영어 타자 게임

```
import java.util.Random;
import java.util.Scanner;
public class TypingGame {
    public static void main(String[] args) {
        String str = "sun moon earth galaxy mountain flower tree bird girl man";
        int n = 1; //문제 번호
        long start, end; //시작, 종료 시간
        double elapsedTime;

        Scanner sc = new Scanner(System.in);
        System.out.print("[타자 게임], 준비되면 엔터!");
        sc.nextLine();
        start = System.currentTimeMillis(); //게임 시작시간

        while(n <= 10) {
            //문자열 잘라서 배열 생성 후 랜덤 추출
            String[] words = str.split(" ");
            Random rnd = new Random();
            int rndIdx = rnd.nextInt(words.length);
            String question = words[rndIdx];
```



영어 타자 게임

● 영어 타자 게임

```
System.out.println("\n문제 번호 " + n);
System.out.println(question); //문제 출제

String answer = sc.nextLine(); //사용자 입력
if(question.equals(answer)) {
    System.out.println("통과!!");
    n++; //다음 문제 번호
}else {
    System.out.println("오타! 다시 도전!");
}
}
end = System.currentTimeMillis(); //게임 종료시간

elapsedTime = (double)(end - start) / 1000;
//서식 문자 사용 : %d-정수, %f-실수, %s-문자열
System.out.printf("게임 소요시간 %.2f초", elapsedTime);

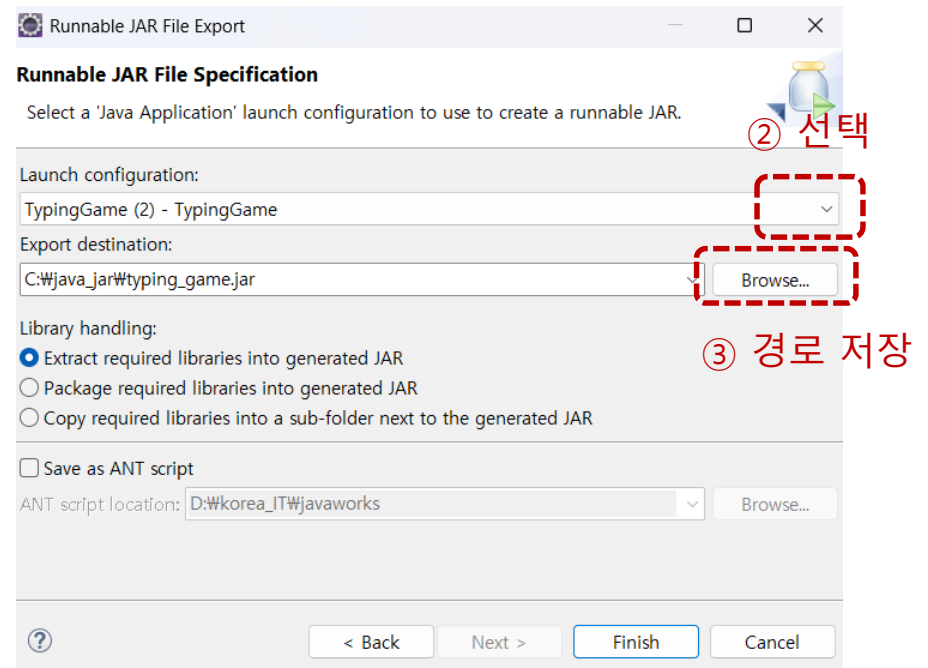
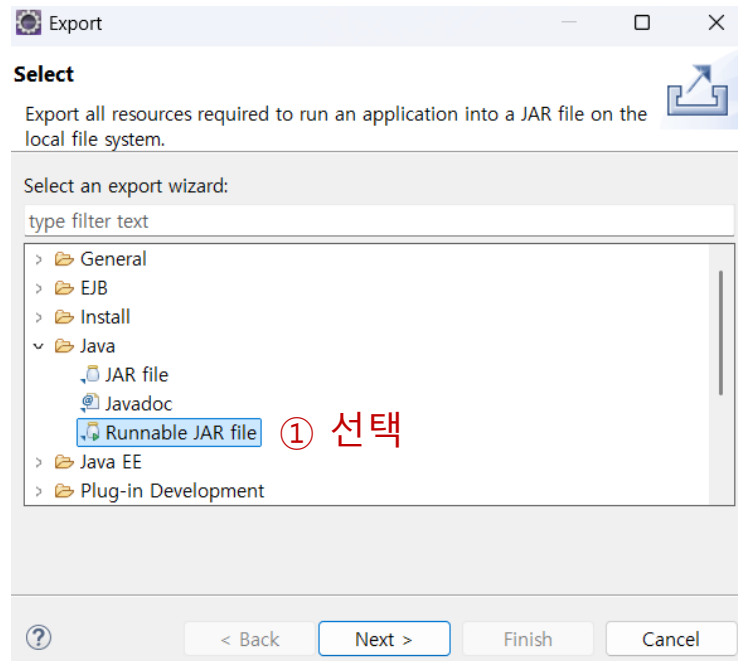
sc.close();
}
}
```



jar 파일 배포

● Jar파일 만들기

프로젝트 이름 – 우측메뉴 – Export – Java – Runnable JAR File



Jar 파일 명령 프롬프트에서 실행

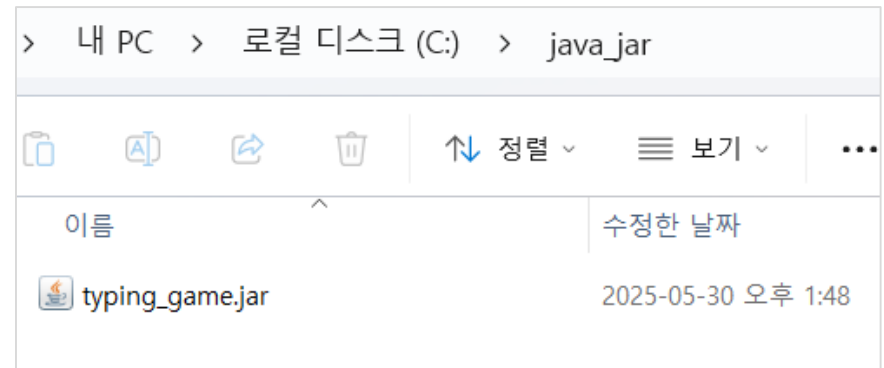
- Jar파일 실행 : `java -jar typing_game.jar`

```
c:\java_jar>java -jar typing_game.jar  
[타자 게임], 준비되면 엔터!
```

```
문제 번호 1  
sun  
sun  
통과!!
```

```
문제 번호 2  
moon  
moon  
통과!!
```

```
문제 번호 3  
tree  
tree  
통과!!
```



실습 문제 1 – 정규 표현식

아래 프로그램을 분석하여 그 실행 결과를 쓰시오.

```
String str = "THANK!@YOU%/";  
String res = str.replaceAll("[^ㄱ-하- | 가-힣a-zA-Z0-9]", "*");  
System.out.println(res);
```

