

12장. java.lang 패키지

1. java.lang 패키지

Java SE에는 다양한 패키지가 존재한다. 그 하나의 java.lang 패키지에는 문자열 조작, 데이터 변환, 수치 연산 스레드 처리 등 기본적인 기능의 API가 포함되어 있다. java.lang 패키지는 Java에서 사용되는 가장 기본적인 클래스를 모은 패키지이다.

일반적으로 패키지의 클래스를 사용하려는 경우 정규화 된 이름에 클래스를 지정하거나 클래스 이름 만 사용하려는 경우에는 패키지를 import 해야 한다. 그러나 Java에서는 java.lang 패키지에 한해 자동으로 import가 된다.

대표적인 java.lang 패키지의 클래스를 다음과 같다.

클래스	설명
object	모든 클래스의 슈퍼 클래스
Math	절대 값, 제곱근 등 다양한 연산을 수행하는 메소드로 이루어짐
System	표준 입력과 표준 출력을 담당하는 클래스
래퍼 클래스	원시 형에 대응한 클래스
Runtime	Java 응용 프로그램 실행의 토대가 되는 객체의 클래스
String	문자열을 다루는 클래스 업데이트 불가
StringBuffer	문자열을 다루는 클래스 업데이트 가능
Class	JavaVM에 로드된 클래스와 인터페이스, 배열 등을 취급
Thread	여러 프로그램을 동시에 실행하는 멀티 스레드 프로그램을 실현
Throwable	예외 클래스 군의 최상위 클래스

이 중 가장 많이 사용하는 **System** 클래스는 다소 특수한 클래스에서 인스턴스화 할 수 없다. 주로 표준 입력과 표준 출력, 표준 오류 출력을 처리하기 위해 필드 나 메소드가 정적으로 정의되어 있다.

2. Object 클래스

java.lang.Object는 Java의 모든 클래스의 수퍼 클래스로 자동으로 모든 클래스가 파생되어진다. 사용자 정의 클래스인 class 라는 키워드를 사용하더라도

묵시적으로 Object의 후손 클래스가 된다. Object 에서 정의 된 메소드는 필요에 따라 상속 된 클래스가 오버라이드 (override)해서 사용하거나 그대로 호출해서 사용하고 있다.

오브젝트의 생성자와 주요 메소드는 다음과 같다

메소드	설명
public String toString()	기본적으로 "개체 이름 + @ + 해시 코드"를 반환한다. 보통은 서브 클래스에서 재정의하고 알기 쉬운 문자열을 반환한다.
public Class getClass ()	리플렉션에 사용됩니다.
protected Object clone()	인스턴스의 복제를 만든다. 필드 (멤버 변수)의 복제 (딥 카피)까지 만들고 싶은 경우, 이 메소드를 오버라이드 하여 자신의 복제를 추가 할 필요가 있다.
public boolean equals(Object)	개체 (값으로) 동일한 경우에 true 를 돌려 준다. 서브 클래스에서 재정의하고 그 클래스에서 상정 비교한다.
public int hashCode ()	해시 코드 값을 돌려 준다. equals()를 직접 정의하는 경우는 해시 코드도 올바르게 반환 할 필요가 있다.
public void wait () public void notify ()	멀티 스레드에 사용된다.
protected void finalize ()	객체가 어디에서도 사용 (참조)되지 않게 하고, GC 에 의해 소멸 될 때 호출되는 메소드.

다음 프로그램으로 코드를 살펴 보자.

```
class Test {  
}  
public class TestToString {  
    public static void main(String[] args) {  
        Test obj1 = new Test();  
        Test obj2 = new Test();  
        Test obj3 = obj1;  
  
        System.out.println(" 연산자로 주소 비교 ");  
        System.out.println("obj1==obj3   " + (obj1 == obj3));  
        System.out.println("obj1==obj2   " + (obj1 == obj2));  
        System.out.println("-----");  
        System.out.println("equals 메소드로 주소비교 ");  
    }  
}
```

```

        System.out.println("obj1.equals(obj3) " + obj1.equals(obj3));
        System.out.println("obj1.equals(obj2) " + obj1.equals(obj2));

        System.out.println("-----");
        System.out.println("obj1의 hashCode : " + obj1.hashCode());
        System.out.println("obj1의 toString : " + obj1.toString());
        System.out.println("obj1의 getClass : " + obj1.getClass());
    }
}

```

[실행결과]

연산자로 주소 비교

obj1==obj3 true

obj1==obj2 false

equals 메소드로 주소비교

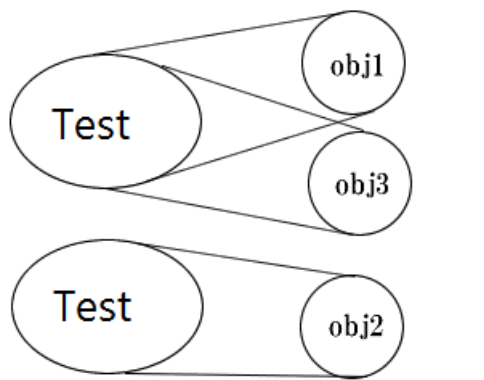
obj1.equals(obj3) true

obj1.equals(obj2) false

obj1의 hashCode :31168322

obj1의 toString :com.sec12.Test@1db9742

obj1의 getClass :class com.sec12.Test












여기에서는 Test 클래스의 객체를 참조 변수 obj1가 참조하고 있다. 또한 Test 클래스를 참조 변수 obj2에서 참조하고 있다. 이후에 참조 변수 obj3에 obj1을 대입하고 있다. 즉, obj1과 obj3 같은 개체를 참조하는 것이다. "=="연산자와 equals 둘 다 주소를 비교하기 때문에 equals 메소드는 재정의 하여 값을 비교하는 메소드로 사용하고 toString()은 해당 클래스의 객체 값을 리턴하는 용도로 재정의 하여 사용한다.

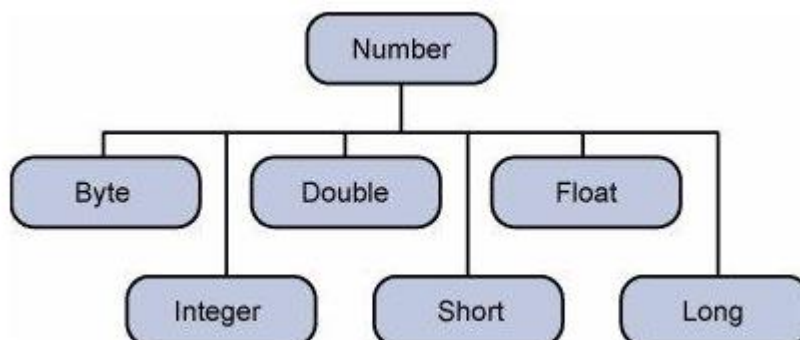
3. Wrapper 클래스

>> Wrapper클래스란?

. Java는 기본 데이터 타입(primitive data type) 형식도 개체로 다룰 수 있도록 각 기본 데이터 형식에 해당하는 변환 클래스가 준비되어 있다. 이러한 클래스를 래퍼 클래스라고 한다 래퍼 클래스는 추상클래스인 `Number` 클래스의 파생클래스로 기본 데이터 타입(primitive data type)에 기능을 추가하여 다른 데이터 형으로 리턴 하는 메서드들로 이루어져 있다.

<<Java Class>>	
	Number
java.lang	
	<u>serialVersionUID: long</u>
	Number()
	intValue():int
	longValue():long
	floatValue():float
	doubleValue():double
	byteValue():byte
	shortValue():short

----->Number클래스가 가진 메소드들을 하위클래스가 재정의하여 사용한다.



다음은 기본 데이터 형이 가진 래퍼클래스이다.

기본 데이터 형	래퍼 클래스	변환 예
boolean	java.lang.Boolean	new Boolean (true); Boolean.valueOf (true);
char	java.lang.Character	new Character ('A'); Character.valueOf ('A')
byte	java.lang.Byte	new Byte ("1"); Byte.valueOf ("1");
short	java.lang.Short	new Short ("1"); Short.valueOf ("1")
int	java.lang.Integer	new Integer (1); Integer.valueOf (1);
long	java.lang.Long	new Long (1L); Long.valueOf (1L);
float	java.lang.Float	new Float (1F); Float.valueOf (1F);
double	java.lang.Double	new Double (1D); Double.valueOf (1D);

공통으로 제공되는 주요 상수로 다음과 같다.

Byte, Short, Integer, Long, Float, Double 에서 제공되는 주요 상수

반환형	상수	설명
static byte static short static int static long static float static double	MAX_VALUE	각 개체 유지하는 설정 가능한 최대 값을 나타냅니다.
static byte static short static int static long static float static double	MIN_VALUE	각 개체 유지하는 설정 가능한 최소치를 나타냅니다.

[Float, Double 에서 제공되는 주요 상수]

반환형	상수	설명
static float static double	NEGATIVE_INFINITY	각 개체의 음의 무한대 값을 나타냅니다.
static float static double	POSITIVE_INFINITY	각 개체 정의의 무한대 값을 나타냅니다.

다음과 같이 공통 상수를 이용하여 각 래퍼의 범위를 확인할 수 있다.

```
public class TestWrapper {
    public static void main(String[] args) {
```

```

System.out.println("byte : "+ Byte.MIN_VALUE+"~"+Byte.MAX_VALUE);
System.out.println("short : "+ Short.MIN_VALUE+"~"+Short.MAX_VALUE);
System.out.println("char:"+(int)Character.MIN_VALUE+"~"+(int)Character.MAX_VALUE);
System.out.println("int : "+ Integer.MIN_VALUE+"~"+Integer.MAX_VALUE);
System.out.println("long: "+ Long.MIN_VALUE+"~"+Integer.MAX_VALUE);
System.out.println("float: "+ Float.MIN_VALUE+"~"+Integer.MAX_VALUE);
System.out.println("double: "+ Double.MIN_VALUE+"~"+Double.MAX_VALUE);
System.out.println("boolean:"+ Boolean.FALSE+"~"+Boolean.TRUE);

    }

}

```

[실행결과]

```

byte : -128~127
short : -32768~32767
char :0~65535
int : -2147483648~2147483647
long: -9223372036854775808~2147483647
float: 1.4E-45~2147483647
double: 4.9E-324~1.7976931348623157E308
boolean:false~true

```

또한 래퍼 클래스이 가진 메소드들 중에는 공통적으로 문자열로 변환할 수 있는 메소드들이 있다
즉, 래퍼 클래스와 String 클래스를 사용하여 형식 변환을 할 수 있다.
변환 형태는 다음과 같다

원래의 형태		변환 된 형태	변환 예
문자열	숫자	byte 형	byte value = Byte.parseByte (string)
		short 형	short value = Short.parseShort (string)
		int 형	int value = Integer.parseInt (string)
		long 형	long value = Long.parseLong (string)
		float 형	float value = Float.parseFloat (string)
		double 형	double value = Double.parseDouble (string)
숫자	문자열	byte 형	String string = String.valueOf (bytevalue)
		short 형	String string = String.valueOf (shortvalue)
		int 형	String string = String.valueOf (intvalue)
		long 형	String string = String.valueOf (longvalue)
		float 형	String string = String.valueOf (floatvalue)
		double 형	String string = String.valueOf (doublevalue)

>>Integer Class

Integer 형의 오브젝트에는 형태가 int 의 단일 필드로 생성되며 int를 String에, String를 int로 변환하는 각종 메소드 나, int의 처리시에 도움이되는 정수 및 메소드도 제공한다.

생성자는 다음과 같다.

Integer (int value)	int 의 인수를 가진 Integer 객체를 생성
Integer (String s)	문자열로 표현되는 수치를 인수로 지정하여 Integer 객체를 생성

주요 메소드는 다음과 같다.

반환 형식	메소드	개요
boolean	equals (Object obj)	이 객체를 지정된 객체와 비교합니다.
int	intValue ()	이 Integer 의 값을 int 값으로
double	doubleValue ()	이 Integer 의 값을 double 값으로 리턴
String	toBinaryString (int i)	2 진수의 값을 문자열로 리턴
String	toHexString (int i)	16 진수의 값을 문자열로 리턴
String	toOctalString (int i)	8 진수의 값을 문자열로 리턴.
int	parseInt (String s)	지정된 문자열을 10 진수의 정수형으로 리턴
int	parseInt (String s, int radix)	지정된 문자열, 표현하고자 하는 진법
String	toString ()	이 Integer 의 값을 나타내는 String 오브젝트 로 리턴
String	toString (int i)	지정된 정수를 나타내는 새로운 String 오브젝트 로 리턴
String	toString (int i, int radix)	2 번째의 인수를 기수로서 1 번째의 인수의 지정된 진법으로 리턴

또한 지정된 값을 다음과 같이 부호없는 정수로 리턴하는 메소드를 가진다 정수로 리턴하는 메소드는 정수형 래퍼클래스인 Byte • Short • Long 에도 같은 유형의 메소드를 가진다.

메소드와 사용예	결과값	설명
Integer.toUnsignedString (0xffffffff)	4294967295	부호없는 정수로 문자열로 리턴
Integer.parseUnsignedInt ("4294967295")	-1	부호없는 정수로 문자열에서 int 로 리턴
Integer.parseUnsignedInt ("+1")	1	
Integer.parseUnsignedInt ("- 1")	NumberFormatException	
Integer.compareUnsigned (0xffffffff, 0x7fffffff)	1	부호없는 정수로 값을 비교
Short.toUnsignedInt ((short) 0xffff)	65535	부호 없음 정수로서 int 로 리턴.
Integer.toUnsignedLong (0xffffffff)	4294967295	부호 없음 정수로서 long 으로 리턴
Integer.divideUnsigned (0xffffffff ,2)	2147483647	부호없는 정수로 나눈 나머지를 리턴
Integer.remainderUnsigned (0xffffffff, 4)	3	부호없는 정수로 나눈 나머지를

		반환한다.
--	--	-------

>>Double 클래스 /Float클래스

Double형의 오브젝트에는 형태가 double의 단일 필드로 생성되며 double를 String에, String 를 double 로 변환하는 각종 메소드 나, double의 처리시에 도움 이되는 메소드도 제공한다.

형태가 float 의 단일 필드로 생성되는 Float 클래스의 주요메소드 또한 Double클래스와 유사하게 사용된다.

주요 메소드는 다음과 같다.

반환 형식	메소드	개요
boolean	equals (Object obj)	이 오브젝트와 지정된 오브젝트를 비교합니다.
byte	byteValue ()	이 Double 의 값을 byte 로 캐스팅하여 byte 로서 돌려줍니다.
short	shortValue ()	이 Double 의 값을 short 로 캐스팅하여 short 로서 돌려줍니다.
int	intValue ()	이 Double 의 정수 값을 int 로 캐스팅하여 반환합니다.
long	longValue ()	long 값을 long 으로 변환하여 반환합니다.
float	floatValue ()	Double 값을 float 로 리턴
double	doubleValue ()	Double 값을 double 로 리턴
boolean	isInfinite ()	Double 값의 절대치가 무한대 값의 경우에 true 를 리턴
boolean	isInfinite (double v)	지정된 수치의 절대치가 무한대 인 경우에는 true 를 리턴
boolean	isNaN ()	이 Double 값이 (NaN) 값의 경우에 true 를 돌려줍니다.
boolean	isNaN (double v)	지정된 수치가 (NaN) 인 경우에 true 를 리턴
double	parseDouble (String s)	Double 클래스의 valueOf 메소드를 실행했을 경우와 같다. String 가 나타내는 값에 초기화 된 새로운 double 값을 리턴한다.
String	toString ()	Double 오브젝트의 String 으로 리턴
String	toString (double d)	double 인수값을 문자열로 리턴

(NaN) -> **Not-a-Number**를 말한다

```
public class TestInfinite {
    public static void main(String[] args) {
        double d1 = 0.5, d2 = 1.0 / 0.0;
```



```
System.out.println ( "d1 :"+ d1);  
System.out.println ( "=>" + Double.isInfinite (d1) );  
System.out.println ( "d2 :"+ d2);  
System.out.println ( "=>" + Double.isInfinite (d2) );→무한대를 체크  
}  
}
```

[실행결과]

```
d1 :0.5  
=>false  
d2 :Infinity  
=>true
```