Wrapper 클래스

Wrapper 클래스

❖ 포장(Wrapper) 객체란?

- 기본 타입(byte, char, short, int, long, float, double, boolean) 값을 내부에 두고 포장하는 객체
- 기본 타입의 값은 외부에서 변경 불가

기본 타입	포장 클래스
byte	Byte
char	Character
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean

Wrapper 클래스

❖ 박싱(Boxing)과 언박싱(Unboxing)

- ㅇ 박싱(Boxing): 기본 타입의 값을 포장 객체로 만드는 과정
- ㅇ 언박싱(Unboxing): 포장 객체에서 기본 타입의 값을 얻어내는 과정

❖ 박싱하는 방법

- 생성자 이용

기본 타입의 값을 줄 경우	문자열을 줄 경우
Byte obj = new Byte(10);	Byte obj = new Byte("10");
Character obj = new Character('가');	
Short obj = new Short(100);	Short obj = new Short("100");
Integer obj = new Integer(1000);	Integer obj = new Integer("1000");
Long obj = new Long(10000);	Long obj = new Long("10000");
Float obj = new Float(2.5F);	Float obj = new Float("2.5F");
Double obj = new Double(3.5);	Double obj = new Double("3.5");
Boolean obj = new Boolean(true);	Boolean obj = new Boolean("true");

Wrapper 클래스

❖ 박싱하는 방법

- ㅇ 각 Wrapper 클래스의 valueOf() 메소드 이용

```
Integer obj = Integer.valueOf(1000);
Integer obj = Integer.valueOf("1000");
```

Wrapper 클래스

❖ 언박싱 코드

- Wrapper 클래스의 내부 값을 기본 데이터타입 값으로 리턴
- 멕서드명 : 기본 타입명 + Value()

기본 타입의 값을 이용		
byte	num	= obj.byteValue();
char	ch	= obj.charValue();
short	num	= obj.shortValue();
int	num	= obj.intValue();
long	num	= obj.longValue();
float	num	= obj.floatValue();
double	num	= obj.doubleValue();
boolean	bool	= obj.booleanValue();

Wrapper 클래스

❖ 기본 타입의 값을 박싱하고 언박싱하기 : BoxingUnBoxingExample.java

```
public class BoxingUnBoxingExample {
    public static void main(String[] args) {
        // Boxing
        Integer obj1 = new Integer(100);
        Integer obj2 = new Integer("200");
        Integer obj3 = Integer.valueOf("300");

        // Unboxing
        int value1 = obj1.intValue();
        int value2 = obj2.intValue();
        int value3 = obj3.intValue();

        System.out.println(value1);
        System.out.println(value2);
        System.out.println(value3);
    }
}
```

❖ 자동 박싱과 언박싱

- 자동 박싱 - 포장 클래스 타입에 기본값이 대입될 경우 발생

```
Integer obj = 100;    //자동 박싱
```

```
List<Integer> list = new ArrayList<Integer>();
list.add(200);   //자동 박싱
```

- 자동 언박싱 - 기본 타입에 포장 객체가 대입될 경우 발생

```
Integer obj = new Integer(200);
int value1 = obj;            //자동 언박싱
int value2 = obj + 100;      //자동 언박싱
```

❖ AutoBoxingUnBoxingExample.java

```
public class AutoBoxingUnBoxingExample {
    public static void main(String[] args) {
        //자동 Boxing
        Integer obj = 100;
        System.out.println("value: " + obj.intValue());

        //대입시 자동 Unboxing
        int value = obj;
        System.out.println("value: " + value);

        //연산시 자동 Unboxing
        int result = obj + 100;
        System.out.println("result: " + result);
    }
}
```

Wrapper 클래스

❖ 문자열을 기본 타입 값으로 변환

- parse + 기본타입 명 → 정적 메소드

기본 타입의 값을 이용		
byte	num	= Byte.parseByte("10");
short	num	= Short.parseShort("100");
int	num	= Integer.parseInt("1000");
long	num	= Long.parseLong("10000");
float	num	= Float.parseFloat("2.5F");
double	num	= Double.parseDouble("3.5");
boolean	bool	= Boolean.parseBoolean("true");

❖ StringToPrimitiveValueExample.java

```
public class StringToPrimitiveValueExample {
    public static void main(String[] args) {
        int value1 = Integer.parseInt("10");
        double value2 = Double.parseDouble("3.14");
        boolean value3 = Boolean.parseBoolean("true");

        System.out.println("value1: " + value1);
        System.out.println("value2: " + value2);
        System.out.println("value3: " + value3);
    }
}
```

Wrapper 클래스

❖ 포장값 비교

- 포장 객체는 내부 값을 비교하기 위해 ==와 != 연산자 사용 불가
- 값을 언박싱해 비교하거나, equals() 메소드로 내부 값 비교할 것

❖ StringToPrimitiveValueExample.java

```
public class ValueCompareExample {
    public static void main(String[] args) {
        System.out.println("[-128~127 초과값일 경우]");
        Integer obj1 = 300;
        Integer obj2 = 300;
        System.out.println("==결과: " + (obj1 == obj2));
        System.out.println("언박싱후 ==결과: " +
                           (obj1.intValue() == obj2.intValue()));
        System.out.println("equals() 결과: " + obj1.equals(obj2));
        System.out.println();

        System.out.println("[-128~127 범위값일 경우]");
        Integer obj3 = 10;
        Integer obj4 = 10;
        System.out.println("==결과: " + (obj3 == obj4));
        System.out.println("언박싱후 ==결과: " +
                           (obj3.intValue() == obj4.intValue()));
        System.out.println("equals() 결과: " + obj3.equals(obj4));
    }
}
```