11장. 기본 API 클래스

이것이 자바다(http://cafe.naver.com/thisjava)

한빛미디어 Hanbit Media, Inc.

7절. String 클래스

❖ String 메소드

- 문자열의 추출, 비교, 찾기, 분리, 변환등과 같은 다양한 메소드 가짐
- 사용 빈도 높은 메소드

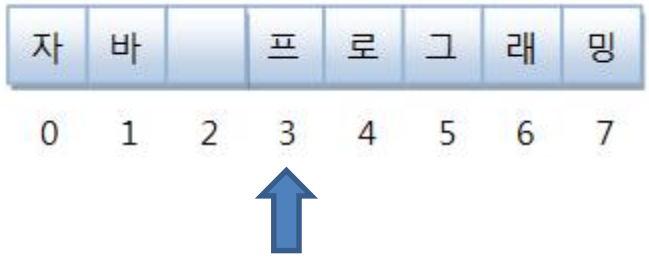
리턴타입	메소드명(매개변수)	설명
char	charAt(int index)	특정 위치의 문자 리턴
boolean	equals(Object anObject)	두 문자열을 비교
byte[]	getBytes()	byte[]로 리턴
byte[]	getBytes(Charset charset)	주어진 문자셋으로 인코딩한 byte[]로 리턴
int	indexOf(String str)	문자열내에서 주어진 문자열의 위치를 리턴
int	length()	총 문자의 수를 리턴
String	replace(CharSequence target, CharSequence replacement)	target 부분을 replacement 로 대치한 새로운 문자열을 리턴
String	substring(int beginIndex)	beginIndex 위치에서 끝까지 짤라낸 새로운 문자열을 리턴
String	substring(int beginIndex, int endIndex)	beginIndex 위치에서 endIndex 전까지 짤라낸 새로운 문자열을 리턴
String	toLowerCase()	알파벳 소문자로 변환한 새로운 문자열을 리턴
String	toUpperCase()	알파벳 대문자로 변환한 새로운 문자열을 리턴
String	trim()	앞뒤 공백을 제거한 새로운 문자열을 리턴
String	valueOf(int i) valueOf(double d)	기본 타입값을 문자열로 리턴

7절. String 클래스

문자 추출(charAt())

- 매개 값으로 주어진 인덱스의 문자 리턴

```
String subject = "자바 프로그래밍";
char charValue = subject.charAt(3);
```



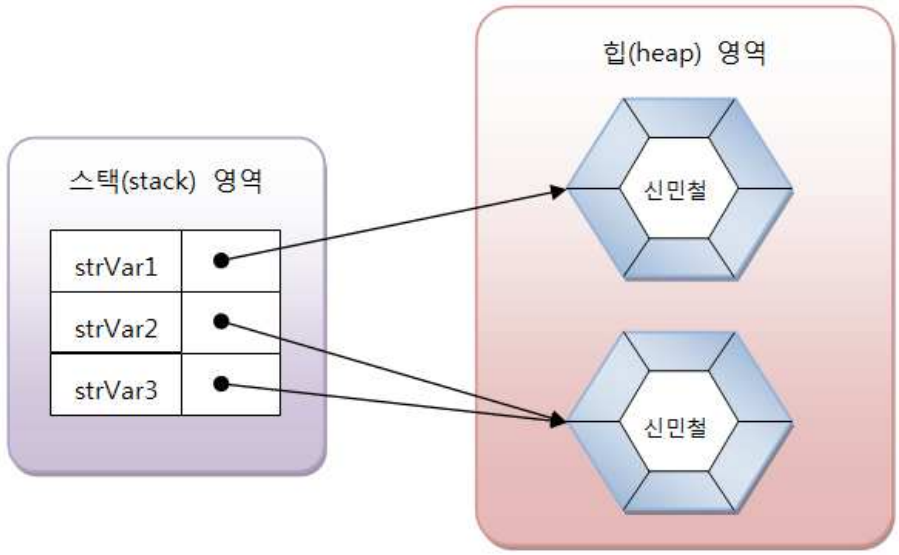
문자열 비교(equals())

- 문자열 비교할 때 == 연산자 사용하면 원하지 않는 결과 발생!

```
String strVar1 = new String("신민철");
String strVar2 = "신민철";
String strVar3 = "신민철";
```

```
strVar1 == strVar2  → false
strVar2 == strVar3  → true
```

```
strVar1.equals(strVar2)  → true
strVar2.equals(strVar3)  → true
```



```
public class StringCharAtExample {
        public static void main(String[] args) {
                String ssn = "010624-1230123";
                char gender = ssn.charAt(7);
                if(gender=='1' || gender=='3')
                {
                        System.out.println("남자 입니다.");
                }
                else if(gender=='2' || gender=='4') {
                        System.out.println("여자 입니다.");

                }
                else {
                        System.out.println("error");
                }
        }
}
```

7절. String 클래스

- 문자열 찾기(indexOf())
  - 매개값으로 주어진 문자열이 시작되는 인덱스 리턴
  - 주어진 문자열이 포함되어 있지 않으면 -1 리턴

```
String subject = "자바 프로그래밍";
int index = subject.indexOf("프로그래밍");
```

  - 특정 문자열이 포함되어 있는지 여부 따라 실행 코드 달리할 때 사용

- 문자열 길이(length()) – 공백도 문자에 포함

```
String subject = "자바 프로그래밍";
int length = subject.length();
```

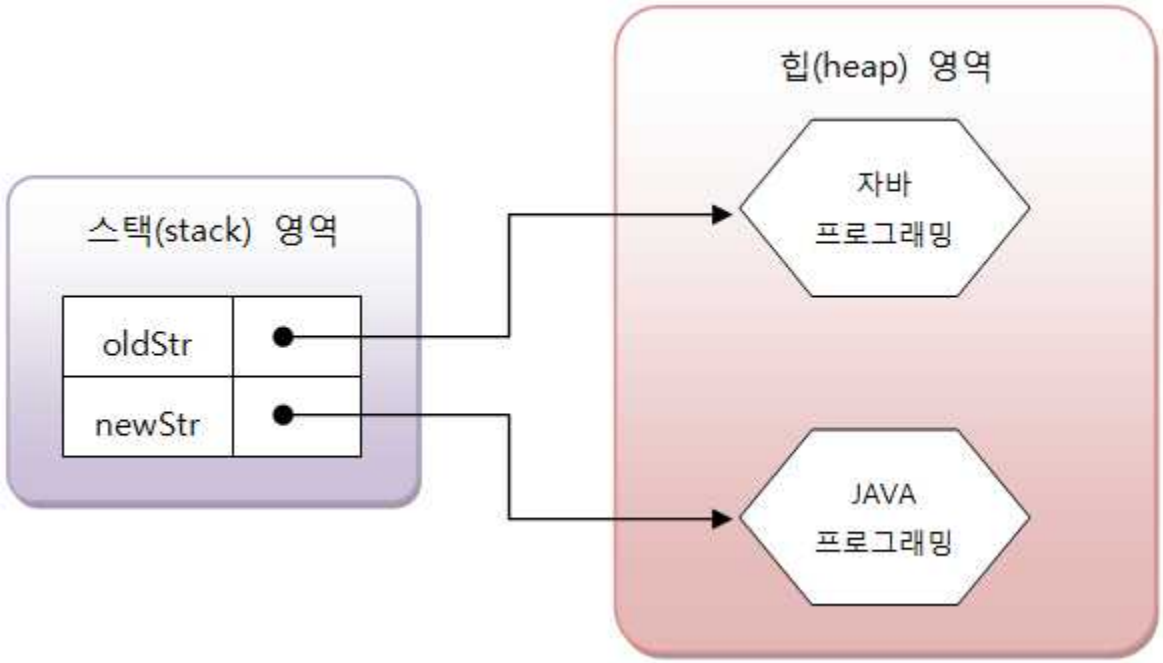
```java
public class StringIndexOfExample {
    public static void main(String[] args) {
        String subject = "자바 프로그래밍";

        int location = subject.indexOf("프로그래밍");
        System.out.println(location);

        if(subject.indexOf("자바") != -1) {
            System.out.println("자바와 관련된 책이군요");
        } else {
            System.out.println("자바와 관련없는 책이군요");
        }
    }
}
```

```java
public class StringLengthExample {
    public static void main(String[] args) {
        String ssn = "7306241230123";
        int length = ssn.length();
        if(length == 13) {
            System.out.println("주민번호 자리수가 맞습니다.");
        } else {
            System.out.println("주민번호 자리수가 틀립니다.");
        }
    }
}
```

7절. String 클래스

- 문자열 대치(replace())
  - 첫 번째 매개값인 문자열 찾음
  - 두 번째 매개값인 문자열로 대치
  - 새로운 문자열 리턴

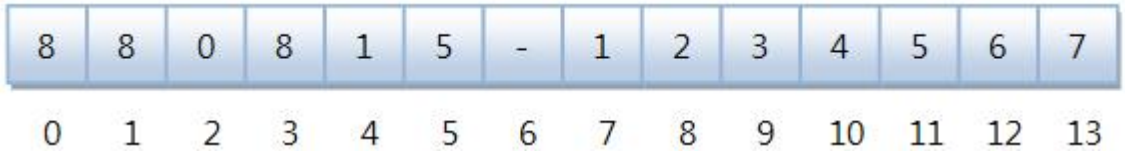
```
String oldStr = "자바 프로그래밍";
String newStr = oldStr.replace("자바", "JAVA");
```



```java
public class StringReplaceExample {
    public static void main(String[] args) {
        String oldStr = "자바는 객체지향언어 입니다. 자바는 풍부한 API를 지원합니다.";
        String newStr = oldStr.replace("자바", "JAVA");

        System.out.println(oldStr);
        System.out.println(newStr);
    }
}
```

7절. String 클래스

- 문자열 잘라내기(substring())
  - substring(int beginIndex, int endIndex)
    - 주어진 시작과 끝 인덱스 사이의 문자열 추출
  - substring(int beginIndex)
    - 주어진 인덱스 이후부터 끝까지 문자열 추출



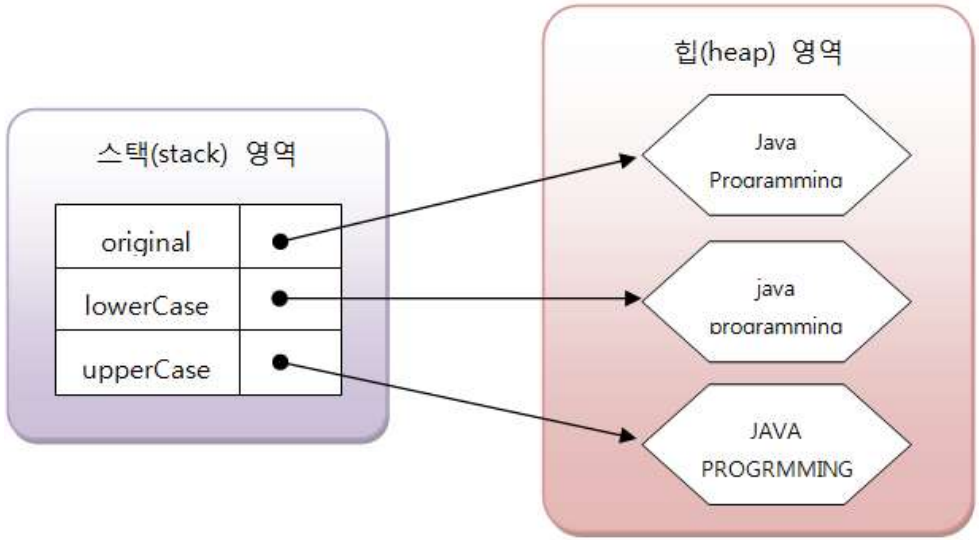

```
String ssn = "880815-1234567";
String firstNum = ssn.substring(0, 6);
String secondNum = ssn.substring(7);
```

```java
public class StringSubstringExample {
    public static void main(String[] args) {
        String ssn = "880815-1234567 ";

        String firstNum = ssn.substring(0, 6);
        System.out.println(firstNum);

        String secondNum = ssn.substring(7);
        System.out.println(secondNum);
    }
}
```

7절. String 클래스

- **알파벳 소·대문자 변경 (toLowerCase(), toUpperCase())**

```
String original = "Java Programming";
String lowerCase = original.toLowerCase();
String upperCase = original.toUpperCase();
```

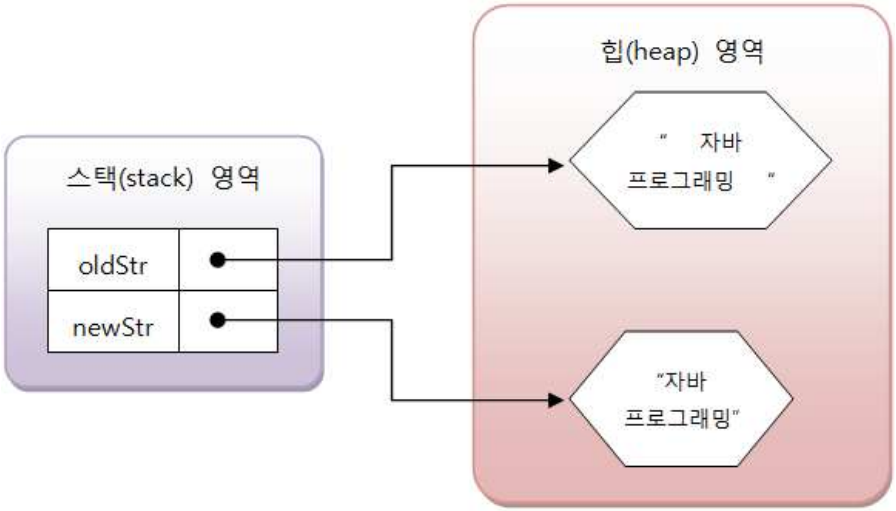


```java
public class StringToLowerUpperCaseExample {
    public static void main(String[] args) {
        String str1 = "Java Programming";
        String str2 = "JAVA Programming";

        System.out.println(str1.equals(str2));

        String lowerStr1 = str1.toLowerCase();
        String lowerStr2 = str2.toLowerCase();
        System.out.println(lowerStr1.equals(lowerStr2));

        System.out.println(str1.equalsIgnoreCase(str2));
    }
}
```

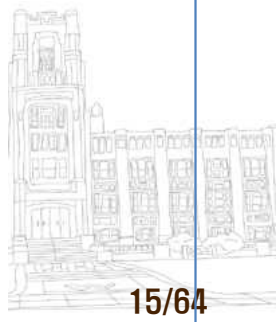
7절. String 클래스

❖ 문자열 앞뒤 공백 잘라내기(trim())

```
String oldStr = "    자바 프로그래밍    ";
String newStr = oldStr.trim();
```



```
public class StringTrimExample {
	public static void main(String[] args) {
		String tel2 = "123   ";
		String tel3 = "   123   ";

		System.out.println("before");//trim을 적용X
		if(tel2.equals(tel3)){
			System.out.println("equals");
		}
		else {
			System.out.println("not equals");
		}

		System.out.println("after");//trim을 적용 , 123만 추출한경우
		tel2=tel2.trim();//"123   " -> "123"
		tel3=tel3.trim();//"   123   " -> "123"
		if(tel2.equals(tel3)){
			System.out.println("equals");
		}
		else {
			System.out.println("not equals");
		}
	}
}
```



7절. String 클래스

- **문자열 변환(valueOf())**
  - **기본 타입의 값을 문자열로 변환**

```
static String valueOf(boolean b)
static String valueOf(char c)
static String valueOf(int i)
static String valueOf(long l)
static String valueOf(double d)
static String valueOf(float f)
```

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

                //숫자 -> 문자열
                String str1 = String.valueOf(10);
                String str2 = String.valueOf(10.5);

                System.out.println(str1);
                System.out.println(str2);

                //문자열 -> 숫자
                int num1= Integer.parseInt(str1);
                double num2= Double.parseDouble(str2);
                System.out.println(num1);
                System.out.println(num2);
        }
}
```

8절. StringTokenizer 클래스

❖ 문자열 분리 방법

- String의 split() 메소드 이용
- java.util.StringTokenizer 클래스 이용

❖ String의 split()

- 정규표현식을 구분자로 해서 부분 문자열 분리
- 배열에 저장하고 리턴

```
홍길동&이수홍,박연수,김자바-최명호
```

```
String[] names = text.split("&|,|-");
```

```java
public class StringSplitExample {
    public static void main(String[] args) {
        String text = "홍길동&이수홍,박연수,김자바-최명호";
        String[] names = text.split("&|,|-");

        for(String name : names) {
            System.out.println(name);
        }
    }
}
```

❖ **$, ^,. 등 일부 특수기호는 사용시 \\를 붙여줘야 함**

```
public class MainApp  {

      public static void main(String[] args)
      {
            String      data = "사과!@#$%^배!@#$%^파인애플";
            String delimiter = "!@#₩₩$%₩₩^";

            String[] dataArr = data.split(delimiter);

            for(int i=0; i<dataArr.length; i++)
            {
                  System.out.println(dataArr[i]);
            }
      }

}
```

8절. StringTokenizer 클래스

❖ StringTokenizer 클래스

```
String text = "홍길동/이수홍/박연수";
StringTokenizer st = new StringTokenizer(text, "/");
```

```
while( st.hasMoreTokens() ) {
   String token = st.nextToken();
   System.out.println(token);
}
```

메소드		설명
int	countTokens()	꺼내지 않고 남아있는 토큰의 수
boolean	hasMoreTokens()	남아 있는 토큰이 있는지 여부
String	nextToken()	토큰을 하나씩 꺼내옴

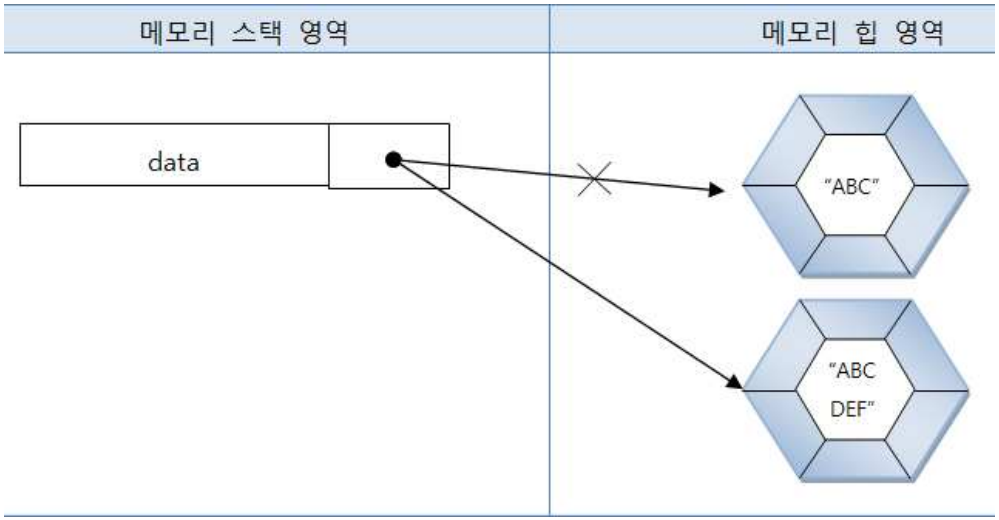
```java
import java.util.StringTokenizer;

public class StringTokenizerExample {
    public static void main(String[] args) {
        String text = "홍길동/이수홍/박연수";

        //how1: 전체 토큰 수를 얻어 for문으로 루핑
        StringTokenizer st = new StringTokenizer(text, "/");
        int countTokens = st.countTokens();
        for(int i=0; i<countTokens; i++) {
            String token = st.nextToken();
            System.out.println(token);
        }

        System.out.println();

        //how2: 남아 있는 토큰를 확인하고 while문으로 루핑
        st = new StringTokenizer(text, "/");
        while( st.hasMoreTokens() ) {
            String token = st.nextToken();
            System.out.println(token);
        }
    }
}
```

9절. StringBuffer, StringBuilder 클래스

❖ 문자열 결합 연산자 +

- String 은 내부의 문자열 수정 불가 -> 대치된 새로운 문자열 리턴

```
String data = "ABC";
data += "DEF";
```



9절. StringBuffer, StringBuilder 클래스

❖ StringBuffer, StringBuilder (p.514~516)

- 버퍼(buffer:데이터를 임시로 저장하는 메모리)에 문자열 저장
- 버퍼 내부에서 추가, 수정, 삭제 작업 가능
- 멀티 스레드환경: StringBuffer 사용
- 단일 스레드환경: StringBuilder 사용

```
StringBuilder sb = new StringBuilder();
StringBuilder sb = new StringBuilder(16);
StringBuilder sb = new StringBuilder("Java");
```

메소드
append(...)
insert(int offset, ...)
delete(int start, int end)
deleteCharAt(int index)
replace(int start, int end, String str)
StringBuilder reverse()
setCharAt(int index, char ch)

```java
public class StringBuilderExample {
    public static void main(String[] args) {
        StringBuilder sb = new StringBuilder();

        sb.append("Java ");
        sb.append("Program Study");
        System.out.println(sb.toString());

        sb.insert(4, "2");
        System.out.println(sb.toString());

        sb.setCharAt(4, '6');
        System.out.println(sb.toString());

        sb.replace(6, 13, "Book");
        System.out.println(sb.toString());

        sb.delete(4, 5);
        System.out.println(sb.toString());

        int length = sb.length();
        System.out.println("총문자수: " + length);

        String result = sb.toString();
        System.out.println(result);
    }
}
```

10절. 정규 표현식과 Pattern 클래스

❖ 정규 표현식(Regular Expression) 작성 방법

- 문자열이 정해져 있는 형식으로 구성되어 있는지 검증할 때 사용
  - Ex) 이메일, 전화번호, 비밀번호 등

- 문자 또는 숫자 기호와 반복 기호가 결합된 문자열

기호	설명		
[ ]	한 개의 문자	[abc]	a, b, c 중 하나의 문자
		[^abc]	a, b, c 이외의 하나의 문자
		[a-zA-Z]	a~z, A~Z 중 하나의 문자
₩d	한 개의 숫자, [0-9]와 동일		
₩s	공백		
₩w	한 개의 알파벳 또는 한 개의 숫자, [a-zA-Z_0-9]와 동일		
?	없음 또는 한 개		
*	없음 또는 한 개 이상		
+	한 개 이상		
{n}	정확히 n 개		
{n,}	최소한 n 개		
{n, m}	n 개에서부터 m 개까지		
( )	그룹핑		

10절. 정규 표현식과 Pattern 클래스

❖ Pattern 클래스

- 정규 표현식으로 문자열을 검증하는 역할
  - 결과는 boolean 타입 !!!

```
boolean result = Pattern.matches("정규식", "입력된 문자열");
```

12절. 포장(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

12절. 포장(Wrapper) 클래스

❖ 박싱(Boxing)과 언박싱(Unboxing)

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

  - valueOf() 메소드 이용

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

```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);
    }
}
```

12절. 포장(Wrapper) 클래스

- 언박싱 코드
  - 각 포장 클래스마다 가지고 있는 클래스 호출
  - 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();

```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);
    }
}
```

12절. 포장(Wrapper) 클래스

❖ 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
```

12절. 포장(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");

❖ 포장값 비교

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

```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);
    }
}
```

13절. Math, Random 클래스

❖ Math 클래스 (예제는 p.533~536 참고)

- 수학 계산에 사용할 수 있는 정적 메소드 제공

메소드	설명	예제 코드	리턴값
int abs(int a) double abs(double a)	절대값	int v1 = Math.abs(-5); double v2 = Math.abs(-3.14);	v1 = 5 v2 = 3.14
double ceil(double a)	올림값	double v3 = Math.ceil(5.3); double v4 = Math.ceil(-5.3);	v3 = 6.0 v4 = -5.0
double floor(double a)	버림값	double v5 = Math.floor(5.3); double v6 = Math.floor(-5.3);	v5 = 5.0 v6 = -6.0
int max(int a, int b) double max(double a, double b)	최대값	int v7 = Math.max(5, 9); double v8 = Math.max(5.3, 2.5);	v7 = 9 v8 = 5.3
int min(int a, int b) double min(double a, double b)	최소값	int v9 = Math.min(5, 9); double v10 = Math.min(5.3, 2.5);	v9 = 5 v10 = 2.5
double random()	랜덤값	double v11 = Math.random();	0.0<= v11<1.0
double rint(double a)	가까운 정수의 실수값	double v12 = Math.rint(5.3); double v13 = Math.rint(5.7);	v12 = 5.0 v13 = 6.0
long round(double a)	반올림값	long v14 = Math.round(5.3); long v15 = Math.round(5.7);	v14 = 5 v15 = 6

```java
public class MathExample {
    public static void main(String[] args) {
        int v1 = Math.abs(-5);
        double v2 = Math.abs(-3.14);
        System.out.println("v1=" + v1);
        System.out.println("v2=" + v2);

        double v3 = Math.ceil(5.3);
        double v4 = Math.ceil(-5.3);
        System.out.println("v3=" + v3);
        System.out.println("v4=" + v4);

        double v5 = Math.floor(5.3);
        double v6 = Math.floor(-5.3);
        System.out.println("v5=" + v5);
        System.out.println("v6=" + v6);

        int v7 = Math.max(5, 9);
        double v8 = Math.max(5.3, 2.5);
        System.out.println("v7=" + v7);
        System.out.println("v8=" + v8);

        int v9 = Math.min(5, 9);
        double v10 = Math.min(5.3, 2.5);
        System.out.println("v9=" + v9);
        System.out.println("v10=" + v10);

        double v11 = Math.random();
        System.out.println("v11=" + v11);

        double v12 = Math.rint(5.3);
        double v13 = Math.rint(5.7);
        System.out.println("v12=" + v12);
        System.out.println("v13=" + v13);

        long v14 = Math.round(5.3);
        long v15 = Math.round(5.7);
        System.out.println("v14=" + v14);
        System.out.println("v15=" + v15);

        double value = 12.3456;
        double temp1 = value * 100;
        long temp2 = Math.round(temp1);
        double v16 = temp2 / 100.0;
        System.out.println("v16=" + v16);
    }
}
```

14절. Date 클래스

❖ Date 클래스

- 날짜를 표현하는 클래스
- 날짜 정보를 객체간에 주고 받을 때 주로 사용

```
Date now = new Date();
String strNow1 = now.toString();
System.out.println(strNow1);

SimpleDateFormat sdf =
        new SimpleDateFormat("yyyy 년 MM 월 dd 일 hh 시 mm 분 ss 초");
String strNow2 = sdf.format(now);
System.out.println(strNow2);
```

【실행 결과】

```java
import java.text.*;
import java.util.*;

public class DateExample {
    public static void main(String[] args) {
        Date now = new Date();
        String strNow1 = now.toString();
        System.out.println(strNow1);

        SimpleDateFormat sdf = new SimpleDateFormat("yyyy년 MM월 dd일 hh시 mm분 ss초");
        String strNow2 = sdf.format(now);
        System.out.println(strNow2);
    }
}
```