java.lang 패키지의 주요 클래스들

학습 목표

- java.lang.Object 클래스의 특징과 자주 사용되는 메소드에 대하여 학습합니다.
- java.lang.String 클래스의 특징과 자주 사용되는 메소드에 대하여 학습합니다.
- wrapper 클래스들의 특징과 종류들에 대하여 학습합니다.
- j2se 5.0 의 Autoboxing / Unboxing 구문에 대하여 학습합니다.

java.lang 패키지의 주요 클래스들

- Object 클래스
- String 클래스
- Wrapper 클래스와 오토박싱(Autoboxing)

Object 클래스

- Object 클래스의 특징
- Object 클래스의 메소드

Object 클래스의 특징

- 자바의 모든 클래스의 최상위 클래스
- extends 키워드 없어도 자동 상속
- Object 클래스의 모든 메소드는 다른 모든 클래스에서 사용 가능

```
class 클래스이름 [extends Object] { ○──────── extends Object는 생략된 상태
    Object 클래스의 메소드 자동 상속 포함
}
```

Object 클래스의 메소드

boolean equals(Object obj)	두 개의 객체가 같은지를 비교하여 같으면 true를, 같지 않으면 false를 리턴
String toString()	현재 객체의 문자열 표현 리턴
protected Object clone()	객체 복사
protected void finalize()	가비지 콜렉션 직전에 객체의 리소스 정리 시 호출
Class getClass()	객체의 클래스 타입 리턴
int hashCode()	객체의 코드 값 리턴
void notify()	wait 된 스레드 실행 재개 시 호출
void notifyAll()	wait 된 모든 스레드 실행 재개 시 호출
void wait()	스레드 일시 중지 시 호출
void wait(long timeout)	주어진 시간만큼 스레드 일시 중지 시 호출
void wait(long timeout, int nanos)	주어진 시간만큼 스레드 일시 중지 시 호출

Object 클래스의 메소드

- equals 메소드
  - 두 객체참조변수가 모두 null이 아니고 참조 객체가 동일하면 true 리턴
  - 하위클래스에서 overriding시에 객체의 타입과 내용으로 객체간의 동등성 비교

```
class MyInt {
    ......
    public boolean equals(Object   obj) {
        if ((obj != null) && (obj instanceof MyInt))
            return  this.value == ((MyInt)obj).value;
        else return false;
    }
    .....
}
```

Object 클래스의 메소드

- toString 메소드
  - 클래스 타입 이름과 객체의 코드값을 16진수로 표현

```
클래스이름@객체의 코드를 나타내는 16진수 값
```

  - System.out.println() 메소드에서 자동 호출
  - 객체의 특정 변수값 리턴하도록 overriding

```
class MyInt {
    int    value;
    ......
    public String toString() {
        return String.valueOf(value);
    }
    ......
}
```

Object 클래스의 메소드

- 프로그램 9-1 실습
  - ObjectTest.java 작성
  - toString()과 equals() 메소드 결과 확인

```
C:₩JAVA>java ObjectTest
MyObject@c17164
MyObject@1fb8ee3
```

Object 클래스의 메소드

- 프로그램 9-2 실습
  - ObjectOverridingTest.java 작성
  - 프로그램 9-1 수정
  - toString()과 equals() 메소드 overriding 결과 확인

```
C:₩JAVA>java ObjectOverridingTest
toString() 메소드 오버라이딩 : MyObject
toString() 메소드 오버라이딩 : MyObject
equals() 비교 true
```

String 클래스

- String 클래스의 특징
- String 클래스의 메소드

String 클래스의 특징

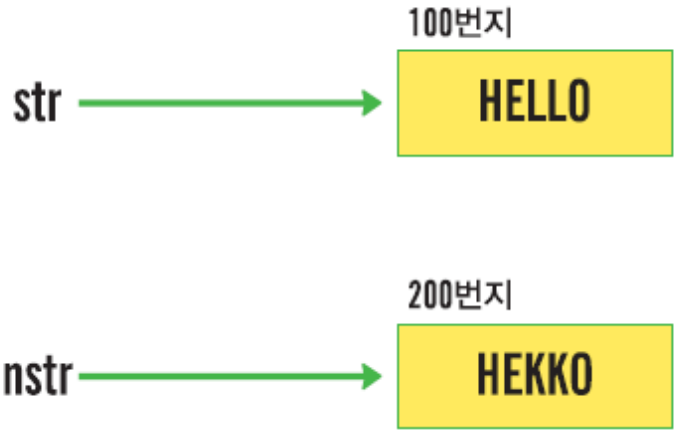
- 문자열 처리 위해 편리한 메소드 제공
- String 클래스의 특징
  - 한 번 만들어진 String 클래스의 객체는 그 값을 변경할 수 없습니다.
  - 이미 동일한 값의 String 리터럴로 생성된 객체가 있다면, 다시 객체를 생성하지 않고 기존의 객체를 참조하기만 합니다.
  - + 연산자를 이용하여 문자열을 결합할 수 있습니다.

String 클래스의 특징

* 한 번 만들어진 String 클래스의 객체는 그 값을 변경할 수 없습니다.

```
String str = new String("HELLO");
String nstr = str.replace('L', 'K');
System.out.println("str = " + str + " new = " + nstr);
```

```
str = HELLO new = HEKKO
```



String 클래스의 특징

- 이미 동일한 값의 String 리터럴로 생성된 객체가 있다면, 다시 객체를 생성하지 않고 기존의 객체를 참조하기만 합니다.

```
String str1 = new String("thanks"); // 라인 1.
String str2 = new String("thanks"); // 라인 2.
if (str1 == str2) System.out.println("str equal");
else System.out.println("str not equal");

String msg1 = "thanks";             // 라인 3.
String msg2 = "thanks";             // 라인 4.

if (msg1 == msg2) System.out.println("msg equal");
else System.out.println("msg not equal");
```

String 클래스의 특징

- + 연산자를 이용하여 문자열을 결합할 수 있습니다.

```
String str1 = "abc" + "def";   // "abcdef"로 문자열 결합
String str2 = 10 + "def";      // int 10을 String 객체 "10"으로 변경한 후에 "10def" 로 문자열 결합
String str3 = "현재시간 : " + new Date();   // Date 객체를 String 객체로 변환하여 "현재시간 : ....."로 문자열 결합
int i1 = 100 + 200;
double d1 = 3.14 + 6.256;      // + 연산자 양쪽에는 int나 double 값의 데이터 존재하는 경우
                               // + 연산자는 문자열 결합이 아니라 정수 또는 실수끼리의 산술 연산
```

String 클래스의 메소드

String 클래스의 메소드

char charAt(int index)	index 번째 문자 리턴
String concat(String str).	+ 연산자처럼 문자열 결합
boolean equals(Objdect anObject)	두 개의 문자열 값 비교하여 true, false 리턴
boolean equalsIgnoreCase (String anotherString)	대소문자를 구분하지 않고 두 개의 문자열 값 비교하여 true, false 리턴
byte[]getBytes()	문자열을 바이트 배열로 변환
byte[]getBytes (String charsetName)	문자열을 지정된 문자셋의 바이트 배열로 변환
void getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin)	문자열의 일부를 문자 배열로 생성
int indexOf(int ch)	ch를 유니코드로 하는 문자의 위치 리턴
int indexOf(int ch, int fromIndex)	fromindex 위치로부터 ch를 유니코드로 하는 문자의 위치 리턴
int indexOf(String str)	str 문자열의 위치 리턴
int indexOf(String str, int fromIndex)	fromindex 위치로부터 str 문자열의 위치 리턴

String 클래스의 메소드

int length()	문자열의 길이 리턴
String replace (char oldChar, char newChar)	oldchar를 newchar로 변환
String substring(int beginIndex)	특정 위치부터의 문자열만 리턴
String substring (int beginIndex, int endIndex)	지정된 특정 위치부터 특정 위치까지의 문자열만 리턴
String toLowerCase()	소문자로 변환
String toUpperCase()	대문자로 변환
String toUpperCase()	대문자로 변환
String trim()	문자열 양쪽의 공백 제거

String 클래스의 메소드

static String valueOf(boolean b)	boolean 데이터형을 문자열로 변환
static String valueOf(char c)	char 데이터형을 문자열로 변환
static String valueOf(char[] data)	문자 배열 데이터형을 문자열로 변환
static String valueOf (char[] data, int offset, int count)	문자 배열의 특정 위치부터 몇 개의 문자만을 문자열로 변환
static String valueOf(double d)	double 데이터형을 문자열로 변환
static String valueOf(float f)	float 데이터형을 문자열로 변환
static String valueOf(int i)	int 데이터형을 문자열로 변환
static String valueOf(long l)	long 데이터형을 문자열로 변환
static String valueOf(Object obj)	객체를 문자열로 변환

String 클래스의 메소드

- toString() 메소드와 equals() 메소드 overriding

```
String s1 = new String ("java");
String s2 = "java";

if (s1.equals(s2)) ←———— s1, s2 모두 String 클래스 타입의 객체이고
                          문자열 “java” 동일. true 리턴
```

String 클래스의 메소드

- 프로그램 9-3 실습
  - StringTest.java 작성
  - String 객체 다양하게 생성
  - == 연산자와 equals() 메소드 차이 확인

```
C:₩JAVA>java StringTest
s1 == s2 equal
s3 == s4 not equal
s1.equals(s2) equal
s3.equals(s4) equal
```

String 클래스의 메소드

- 프로그램 9-4 실습
  - StringMethodTest.java 작성
  - String 클래스의 다양한 메소드 이용

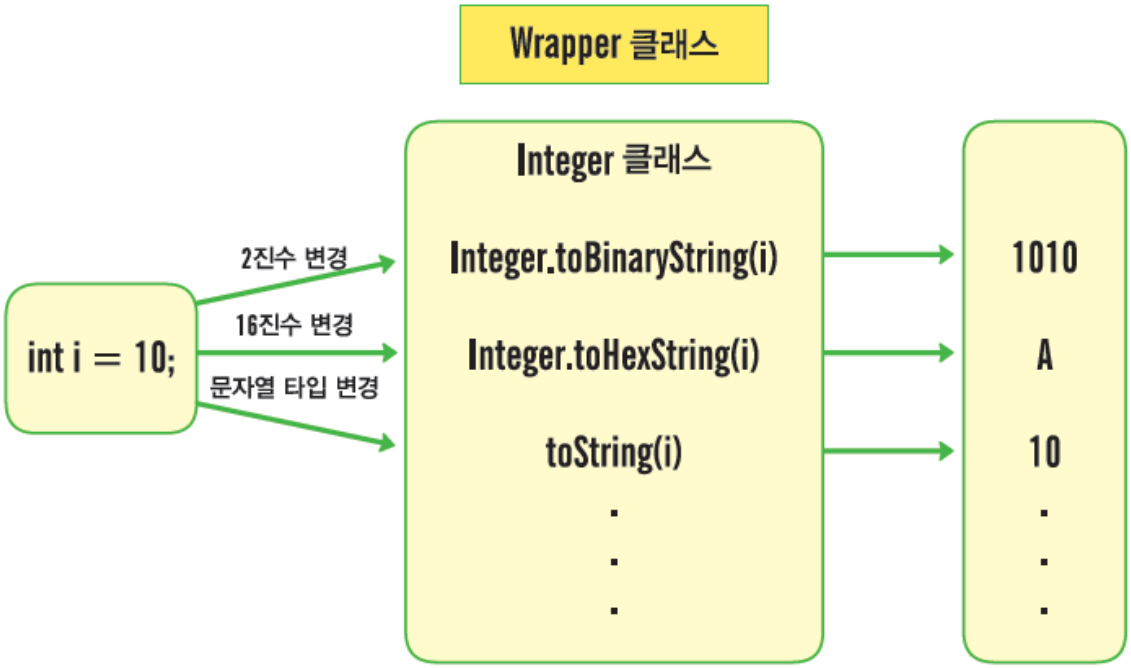
```
C:₩JAVA>java StringMethodTest
s1's Length : 5
s2's Length : 9
s1과 s2를 결합 : JAVA PROGRAM!
s1의 공백을 없애면 JAVA
s1을 모두 소문자로 바꾸면java
소문자로 변경 이후의 s1 : JAVA
s3의 문자열 대체 : Hekko
문자열 대체 이후의 s3 : Hello
```

Wrapper 클래스와 오토박싱(Autoboxing)

- Wrapper 클래스란
- Wrapper 클래스의 종류
- Autoboxing / unboxing

Wrapper 클래스란

- 기본형 변수에 적용 가능한 연산자로 제공되지 않는 기능 구현, 제공하는 클래스
- 포장 클래스 또는 Wrapper 클래스라 함



Wrapper 클래스의 종류

* Wrapper 클래스의 종류

Wrapper 클래스 이름	기본형 변수 타입
Boolean	boolean
Character	char
Byte	byte
Short	short
Integer	int
Long	long
Float	float
Double	double

Wrapper 클래스의 종류

* Wrapper 클래스 객체 생성

```
기본형_변수_타입 변수이름 = 데이터값;
Wrapper_클래스 객체참조변수이름 = new Wrapper_클래스(변수이름);
```

```
boolean b = true;
Boolean wb = new Boolean(b);

int i = 100;
Integer wi = new Integer(i);
```

* toString() 메소드와 equals() 메소드 overriding

```
Boolean b1 = new Boolean(true);
Boolean b2 = new Boolean(true);

if (b1.equals(b2))
===> b1, b2 모두 Boolean 클래스 타입의 객체이고 데이터 값 동일. true 리턴.

Integer i1 = new Integer( 9 );
Integer i2 = new Integer( 9 );

if(i1.equals(i2))
===> i1, i2 모두 Integer 클래스 타입의 객체이고 데이터 값 동일. true 리턴.
```

Wrapper 클래스의 종류

- 프로그램 9-5 실습
  - WrapperTest.java 작성
  - Wrapper 클래스 객체 생성하여 다양한 메소드 이용

```
C:₩JAVA>java WrapperTest
Boolean 객체 : true
Boolean 객체 : true
Integer 객체 : 10
Double 객체 : 10.0
int의 최대값 : 2147483647
int의 최소값 : -2147483648
같다
a를 대문자로 변환 : false
-1의 이진수 : 11111111111111111111111111111111
String 3.14 를 실수로 변환 : 3.14
```

Autoboxing / unboxing

- J2SE 5.0 버전의 새로운 기능
- AutoBoxing은 기본형 변수를 대응되는 Wrapper 클래스의 객체로 대입할 때 컴파일러가 자동으로 Wrapper 클래스 타입으로 변환해주는 기능

```
int i = 20;
Integer in = i;  ——— 두 번째 문장은 컴파일러가 자동으로 아래 문장과 같이 처리합니다.
Integer in = new Integer(i);
```

```
int i = 20;
Integer in = i;
int val = in;  ——— 세 번째 문장은 컴파일러가 자동으로 아래 문장과 같이 처리합니다.
int val = in.intValue();
```

Autoboxing / unboxing

- 자바 언어가 기본형 변수와 참조형 변수 간에 데이터 타입 변환이 가능하도록 변경된 것은 아님
- AutoBoxing/Unboxing은 자바 언어를 이용한 “개발의 편리성”의 한 단면

Autoboxing / unboxing

- 프로그램 9-6 실습
  - AutoboxingTest.java 작성
  - autoboxing 이용하여 기본형 변수를 객체 형태로 사용

```
C:₩JAVA>java AutoboxingTest
i1 == i2 true
i1.equals(i2) true
10
10
10
```

정리

- Object 클래스는 자바의 모든 클래스에서 최상위 클래스로 모든 클래스 타입의 객체들이 사용 가능한 메소드들을 모아 놓은 유틸리티 클래스입니다.
- Object 클래스의 toString() 메소드는 객체의 특정 값을 출력하고자 할 때 overriding 하여 사용할 수 있습니다.
- Object 클래스 내의 equals() 메소드는 두 객체 간의 동등성을 비교하고자 하는 경우 overriding하여 사용할 수 있습니다.

정리

- String 클래스는 문자열을 표현하고 조작하기 위해 사용합니다.
- String 클래스는 다음의 특징을 가지는 클래스입니다.
  - 한 번 만들어진 String 클래스의 객체는 그 값을 변경할 수 없습니다.
  - String 리터럴로 객체를 생성하려 할 때, 만약 이미 동일한 값의 String 리터럴로 생성된 객체가 있다면, 다시 객체를 생성하지 않고 기존의 객체를 참조하기만 합니다.
  - + 연산자를 이용하여 문자열을 결합할 수 있습니다.

정리

- Wrapper 클래스는 기본형 변수에 적용할 수 있는 연산자로 제공되지 않는 기능을 메소드로 구현한 클래스입니다.

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

정리

* Wrapper 클래스 타입의 객체를 생성할 때에는 각 Wrapper 클래스와 매핑 가능한 기본형 변수 값을 매개변수로 전달받습니다.

```
기본형_변수_타입 변수이름 = 데이터 값;
Wrapper_클래스 객체참조변수이름 = new Wrapper_클래스 (변수이름);
```

* J2SE 5.0에서 추가된 Autoboxing / unboxing 구문은 Wrapper 클래스와 기본형 변수 간의 데이터 타입 변환을 쉽게 도와주는 구문입니다.