

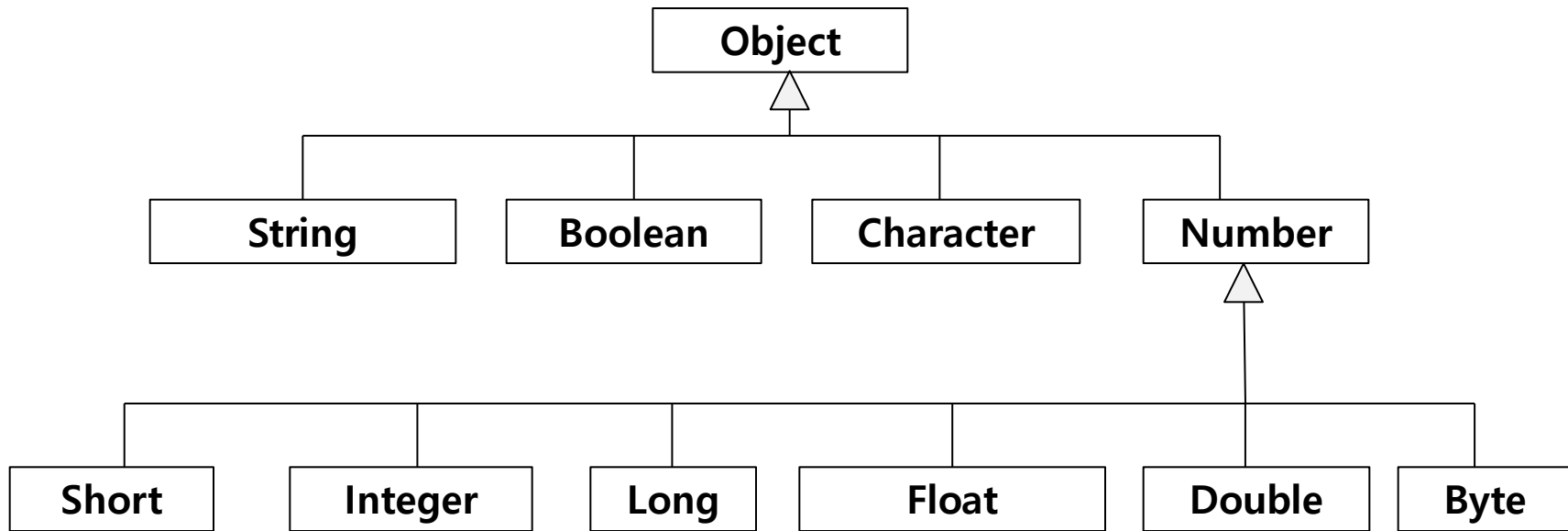
chapter03

자바 기본 API

1. 개요
2. Object Class
3. String Class
4. Wrapper Class

■ 개요

- 자바프로그램에서 가장 많이 사용되는 패키지
- Import 를 사용하지 않아도 자동으로 포함



chapter03

자바 기본 API

1. 개요
2. **Object Class**
3. String Class
4. Wrapper Class

■ Object Class

- 모든 클래스의 최상위 클래스
- 명시적 extends java.lang.Object 없이도 자동으로 상속 받게 된다.

PointApp.java

```
public class PointApp {  
  
    public static void main( String[] args ) {  
        Point a = new Point(2,3);  
  
        System.out.println(a.hashCode());  
        System.out.println(a.getClass());  
        System.out.println(a.getClass().getName());  
        System.out.println(a.toString());  
        System.out.println(a);  
    }  
}
```

Object.java

```
hashCode();  
getClass().getName();  
toString();  
equals();
```

Point.java

```
public class Point {  
    private int x;  
    private int y;  
    public Point( int x, int y ) {  
        this.x = x;  
        this.y = y;  
    }  
}
```



■ toString()

- Object의 toString : getClass().getName() + "@" + Integer.toHexString(hashCode())

Object.java

```
hashCode();  
getClass().getName();  
toString();  
equals();
```

Point.java

```
public class Point {  
    private int x;  
    private int y;  
  
    public Point( int x, int y ) {  
        this.x = x;  
        this.y = y;  
    }  
  
    public String toString(){  
        return "Point(" + x + "," + y + ")"; // Point 클래스만의 toString()  
    }  
}
```

Problems @ Javadoc Decla
<terminated> PointApp (2) [Java Appli
Point(2,3)

■ equals()

- 두 객체의 비교시 == 와 Object 클래스의 equals() 메소드를 사용한다.
- == 와 equals()의 상당한 차이점
 - ✓ == 참조변수값 비교
 - ✓ equals() 정의한 값 비교 → 재정의 해서 사용
- 참조변수값을 먼저 비교한다.
참조변수값이 같으면 두 객체는 같은것으로 한다.
참조변수값이 다르면 두 객체의 속성값을 비교한다.

Point.java

```
public class Point {  
    private int x;  
    private int y;  
    ...  
  
    //equal메소드 재정의  
    public boolean equals(Object obj) {  
        Point p = ((Point)obj);  
        if(this.hashCode()==p.hashCode()){  
            return true;  
        }else if( this.x == p.x && this.y == p.y) {  
            return true;  
        } else {  
            return false;  
        }  
    }  
}
```

[문제]

- Rectangle 클래스를 만들고 equals() 사용해 봅시다.
- int 타입의 width, height의 필드를 가지는 Rectangle 클래스를 작성하고 인스턴스를 생성합니다.
- 구해지는 면적이 같으면 두 객체가 같은 것으로 판별하도록 equals()를 오버라이딩 하세요.

RectangleApp.java

```
public class RectangleApp{  
    public static void main(String[] args){  
  
        Rectangle a = new Rectangle(6,4);  
        Rectangle b = new Rectangle(2,12);  
        Rectangle c = new Rectangle(3,3);  
        Rectangle d = c;  
  
        System.out.println(a.equals(b));  
        System.out.println(a.equals(c));  
        System.out.println(a.equals(d));  
        System.out.println(d.equals(c));  
  
    }  
}
```

chapter03

자바 기본 API

1. 개요
2. Object Class
3. **String Class**
4. Wrapper Class

03 String Class

■ 문자(Character)

- 단순 자료형(char)
- 문자 선언 `char letter;`
- 문자열 할당 `letter = 'A';`
- 문자열 사용 `System.out.println(letter);`

■ 문자열(String) 클래스

- 연속된 문자(character)들을 저장하고 다루기 위한 클래스
- 문자열 상수: " "로 둘러싸인 문자열 → 한번 만들어진 String 객체는 변경 불가함(immutable)
- 문자열 선언 `String greeting;`
- 문자열 할당 `greeting = "Hello JAVA!";`
- 문자열 사용 `System.out.println(greeting);`
- 특수 문자의 표현(Escape characters)
 - `System.out.print("Hello \"JAVA!\"");`
 - `System.out.print("Hello \\\"JAVA!\\\"");`
 - `System.out.print("Hello \\\"JAVA!\\\"\\n");`

03 String Class

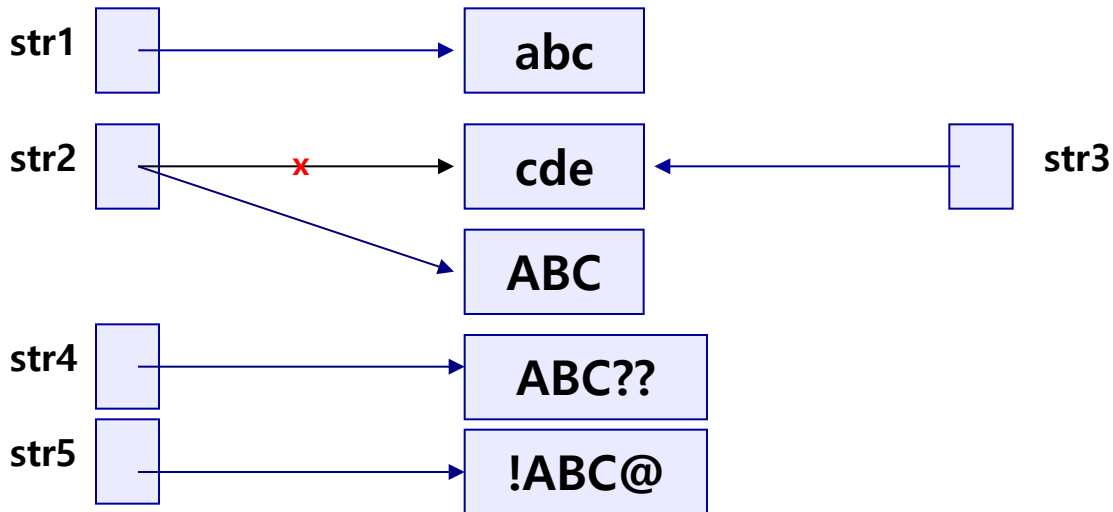
■ String 클래스- 메모리

```
String str1;  
String str2, str3; // String 클래스변수 str1, str2, str3 선언
```

```
str1 = "abc";      // str1은 생성된 String 클래스의 객체(Object)를 가리킴  
str2 = "cde";      // str2은 생성된 String 클래스의 객체(Object)를 가리킴  
str3 = str2;       // str3에 str2의 값 할당
```

```
str2 = str1.toUpperCase();  
String str4 = str2.concat( "??" );  
String str5 = "!".concat(str2).concat( "@" );
```

```
System.out.println("str1: " + str1);  
System.out.println("str2: " + str2);  
System.out.println("str4: " + str4);  
System.out.println("str5: " + str5);
```



03 String Class

■ String 클래스 연산

- + 연산자

```
String greet = "Hello";
```

```
String name = "JAVA";
```

```
System.out.println( greet + name + "!" );
```

```
System.out.println( greet + " " + name + "!" );
```

- index

- String 객체 내의 문자 인덱싱은 배열과 같이 0부터 시작됨

- `charAt(position)` : 해당 위치의 문자를 반환

- `substring(start, end)` : start부터 end까지의 문자들을 새로운 문자열로 반환

```
String greeting = "Hello JAVA!";
```

```
greeting.charAt( 0 );
```

```
greeting.charAt( 10 );
```

```
greeting.substring( 1, 3 );
```

Index	0	1	2	3	4	5	6	7	8	9	10
char	H	e	l	l	o		J	A	V	A	!

03 String Class

■ 실습예제

chapter03

자바 기본 API

1. 개요
2. Object Class
3. String Class
4. **Wrapper Class**

04 Wrapper Class

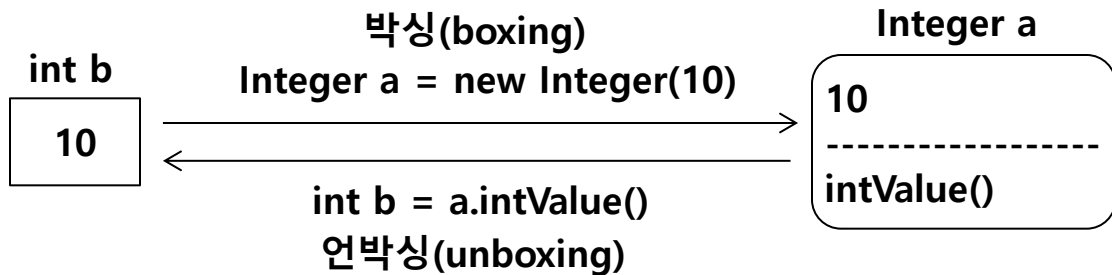
■ Wrapper Class

- 특정 클래스를 가리키는 이름이 아님
- 8개의 기본데이터형을 객체형식으로 다루기 위한 클래스들의 통칭
 - ✓ byte → Byte
 - ✓ short → Short
 - ✓ int → Integer
 - ✓ long → Long
 - ✓ char → Character
 - ✓ float → Float
 - ✓ double → Double
 - ✓ boolean → Boolean
- 사용하는 이유
 - 자바 세상에는 객체만 있기 때문에 객체를 대상으로 처리하는 경우가 많음
 - 어떤 클래스는 객체만을 다루기 때문에 기본 데이터형을 쓸 수 없다.

04 Wrapper Class

■ Wrapper 클래스 박싱(boxing)과 언박싱(unboxing)

- 기본 데이터 타입을 Wrapper 클래스로 변환하는 것을 boxing 이라 한다.
- 반대의 경우를 unboxing 이라 한다.



- JDK1.5 이상부터는 박싱(자동박싱)과 언박싱(자동언박싱)이 자동으로 일어남