제5장 클래스와 객체

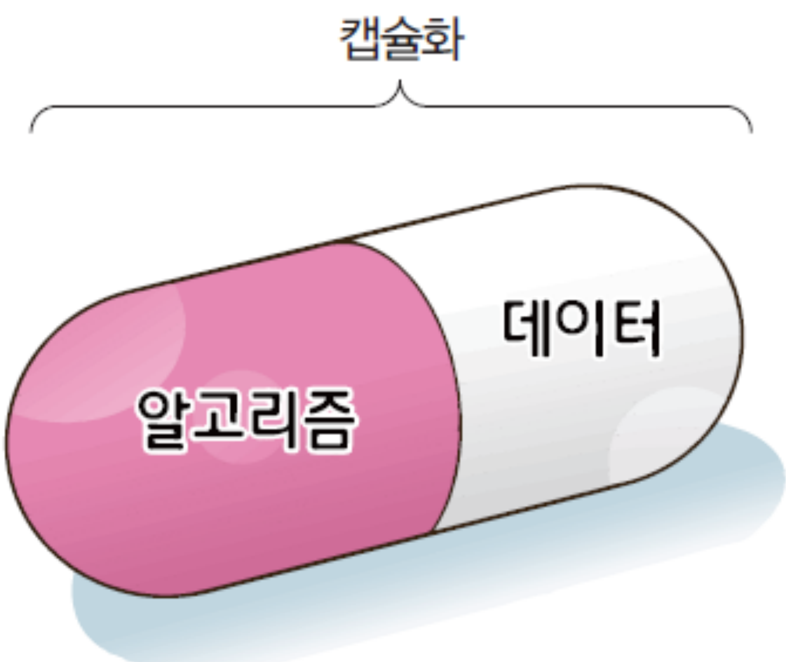
- 객체 지향 특징
- 변수의 종류
- 문자열 관련 함수

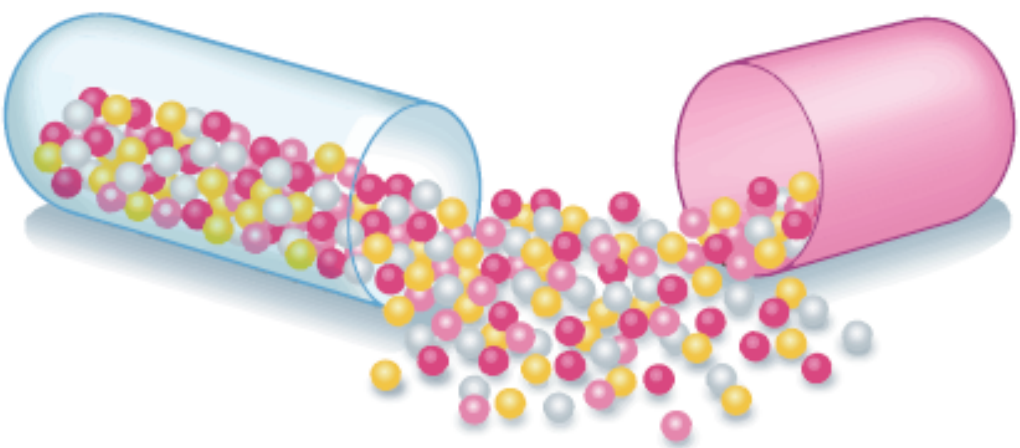
객체 지향의 3대 특징

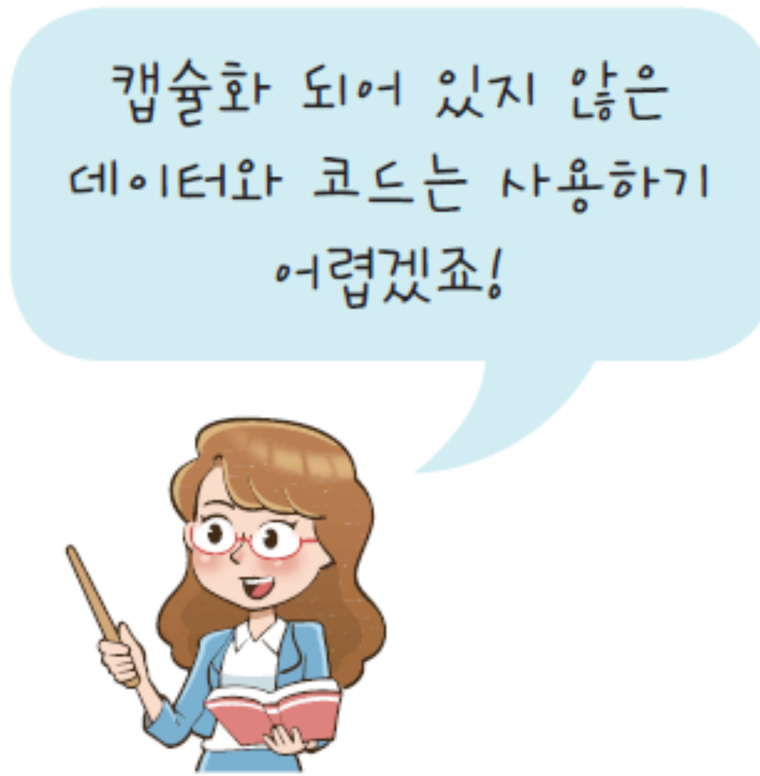
- 캡슐화
- 상속
- 다형성

캡슐화

- **캡슐화(encapsulation)**: 관련된 데이터와 알고리즘(코드)이 하나의 묶음으로 정리되어 있는 것

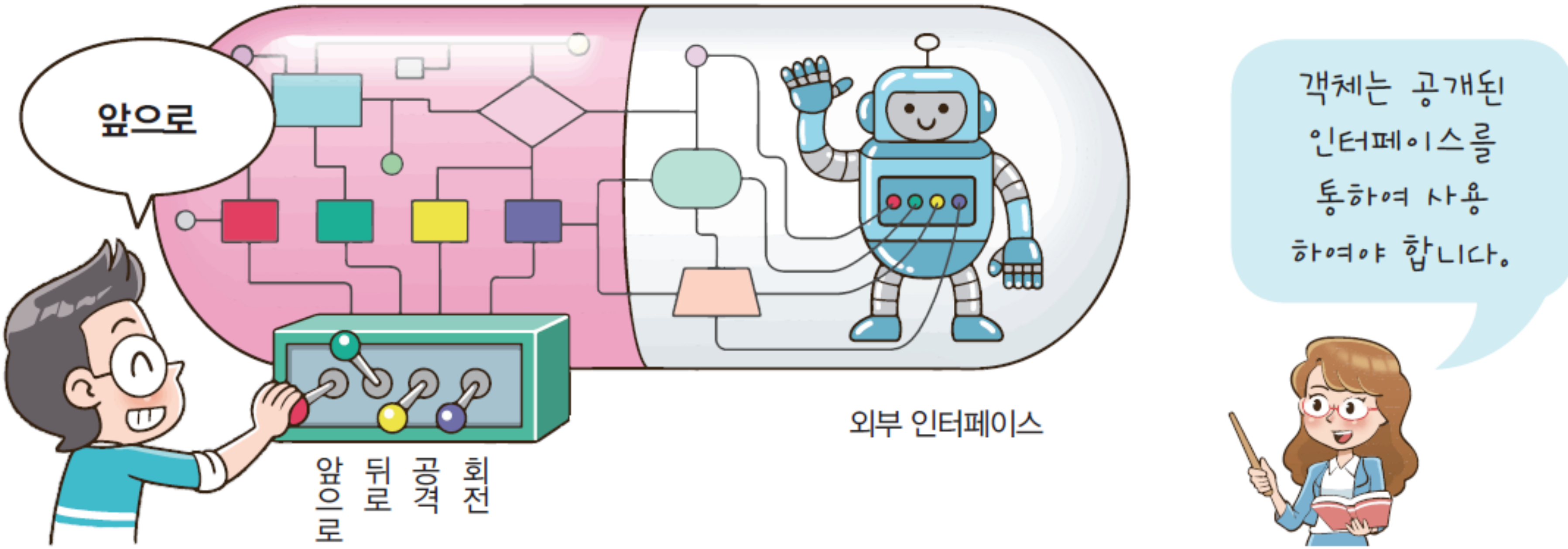




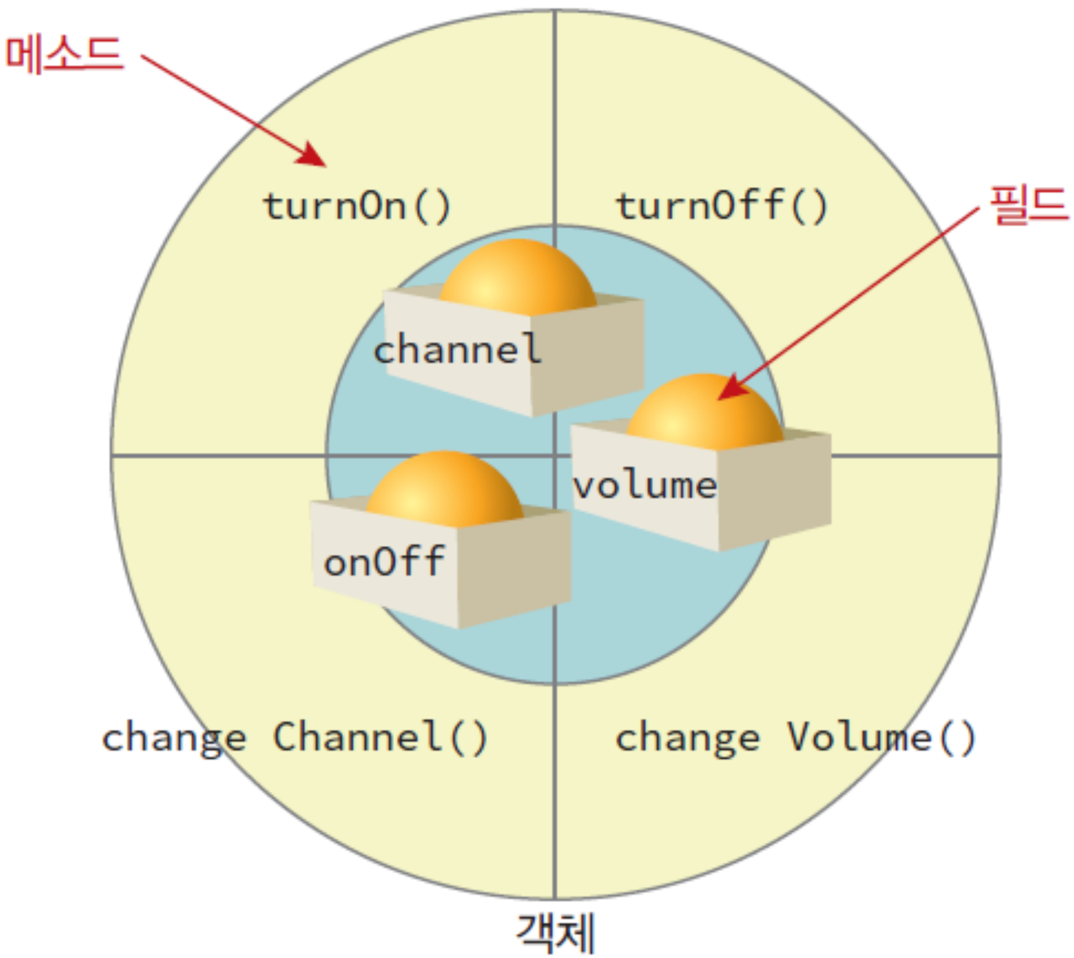

캡슐화 되어 있지 않은
데이터와 코드는 사용하기
어렵겠죠!

캡슐화와 정보 은닉

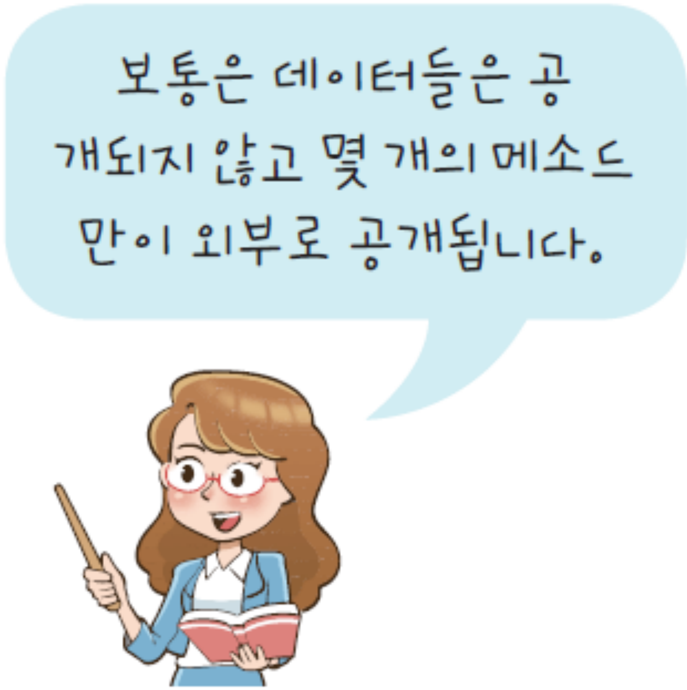
- 정보 은닉(information hiding)은 객체를 캡슐로 싸서 객체의 내부를 보호하는 하는 것
- 객체의 실제 구현 내용을 외부에 감추는 것



캡슐화와 정보 은닉



메소드
turnOn()
turnOff()
필드
channel
volume
onOff
change Channel()
change Volume()
객체

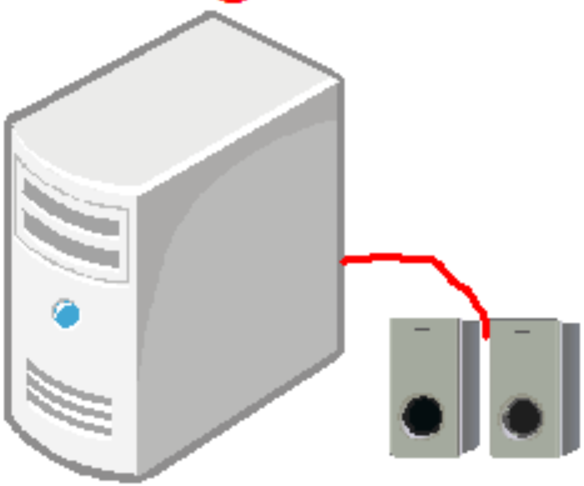


보통은 데이터들은 공개되지 않고 몇 개의 메소드만이 외부로 공개됩니다.

업그레이드가 쉽다.

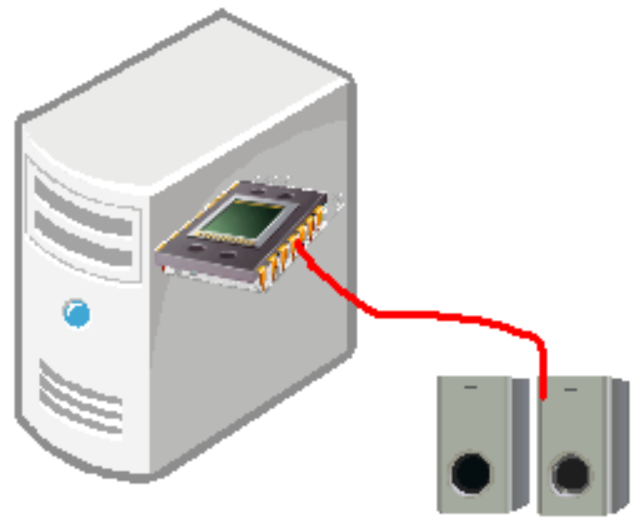
- 라이브러리가 업그레이드되면 쉽게 바꿀 수 있음
- 정보 은닉이 가능하기 때문에 업그레이드 가능

O



만약 외부의 표준 오디오 단자를 이용하였으면 내부의 사운드 카드를 변경할 수 있다.

X



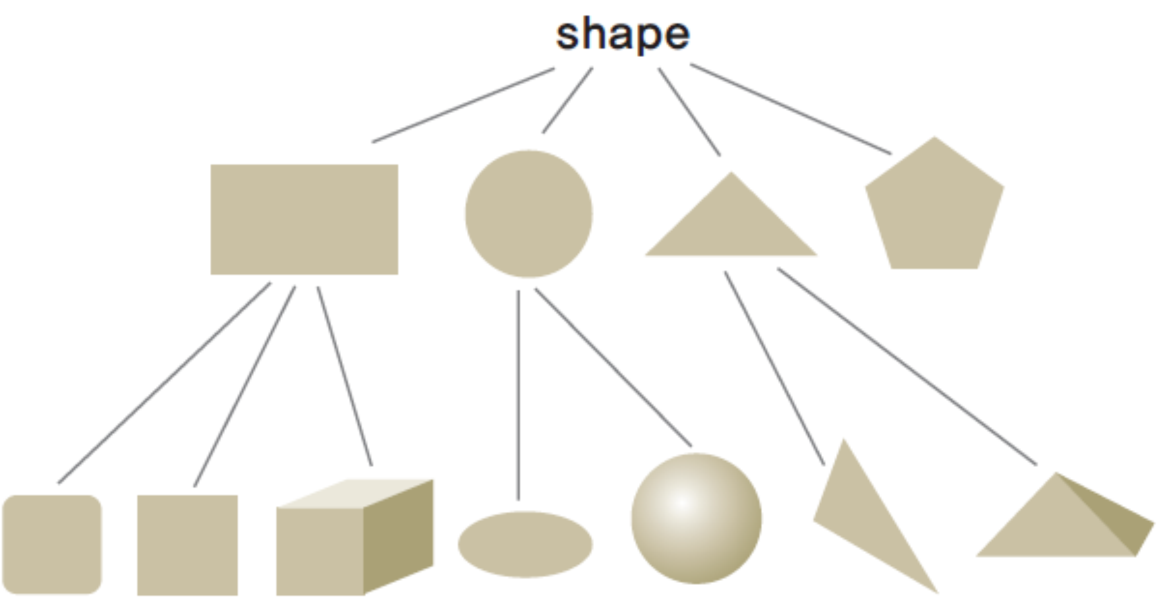
만약 내부의 오디오 제어 칩의 단자에 연결하였으면 내부의 사운드 카드를 변경할 수 없다.

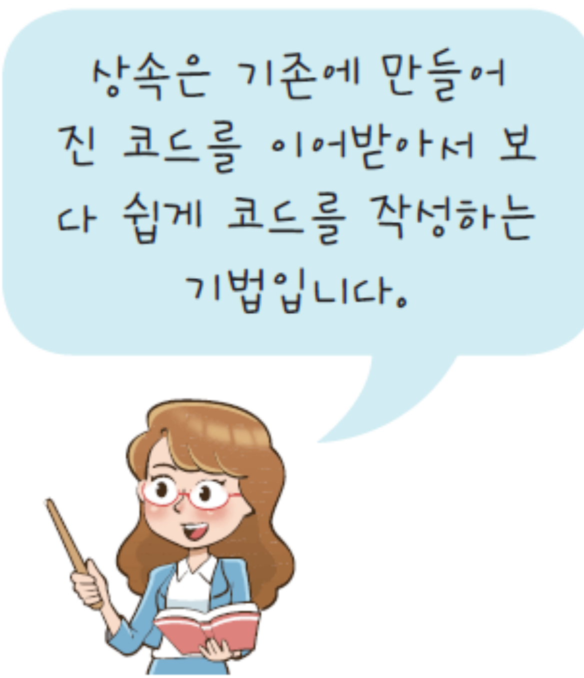
상속

- 상속(inheritance)
  - 이미 작성된 클래스(부모 클래스)를 이어받아서 새로운 클래스(자식 클래스)를 생성하는 기법

상속

- 기존의 코드를 재활용하기 위한 기법





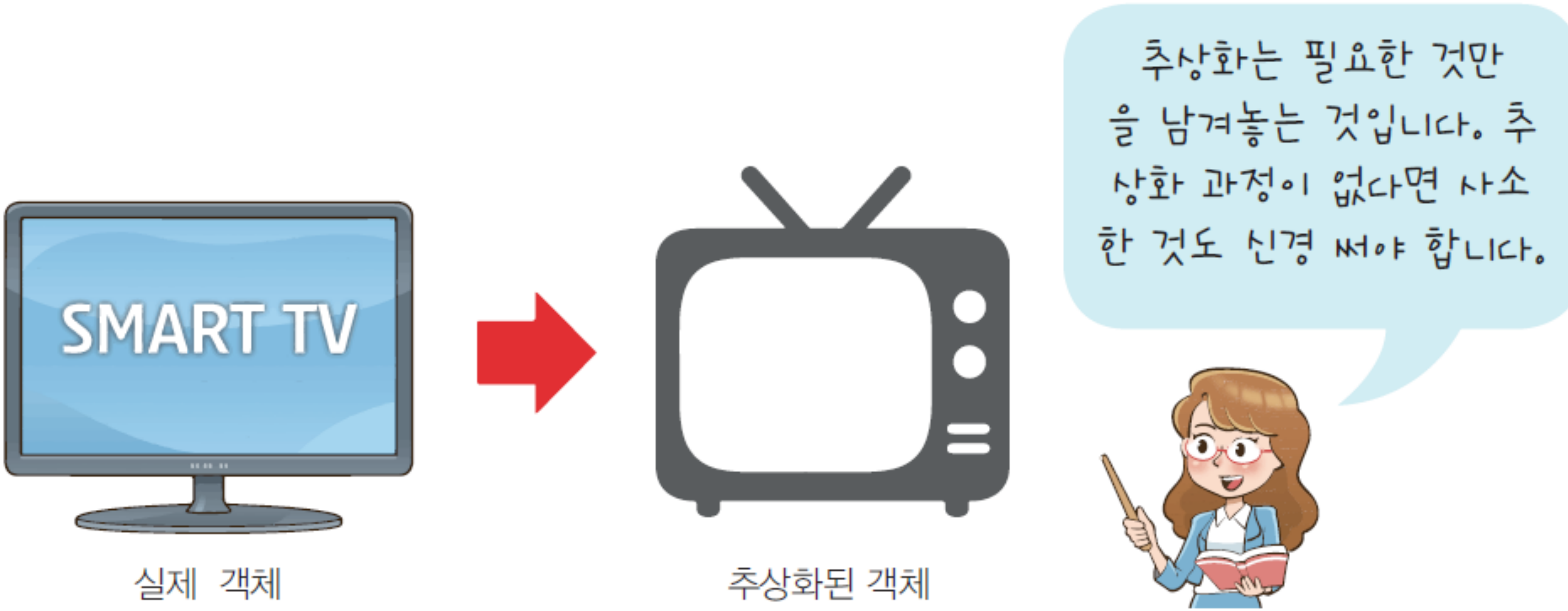
다형성

- 하나의 이름(방법)으로 많은 상황에 대처하는 기법
  - 개념적으로 동일한 작업을 하는 멤버 함수들에 똑같은 이름을 부여할 수 있다
  - 그러므로 코드가 더 간단해진다

다형성

- 하나의 이름(방법)으로 많은 상황에 대처하는 기법

추상화


SMART TV
실제 객체
추상화된 객체
추상화는 필요한 것만을 남겨놓는 것입니다. 추상화 과정이 없다면 사소한 것도 신경 써야 합니다.

객체 지향의 장점

- 신뢰성있는 소프트웨어를 쉽게 작성할 수 있다.
- 코드를 재사용하기 쉽다.
- 업그레이드가 쉽다.
- 디버깅이 쉽다.

쉬운 디버깅

- 예를 들어서 절차 지향 프로그램에서 하나의 변수를 1000개의 함수가 사용하고 있다고 가정해보자.

- -> 하나의 변수를 1000개의 함수에서 변경할 수 있다.

쉬운 디버깅

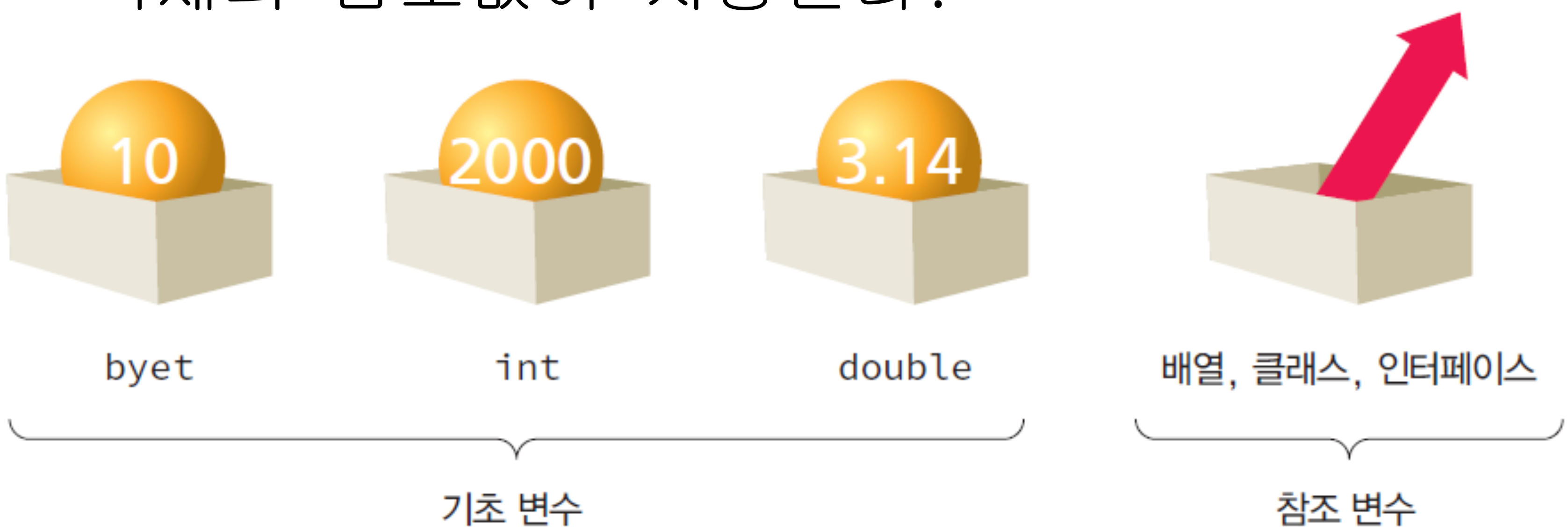
- 객체 지향 프로그램에서 100개의 클래스가 있고 클래스당 10개의 메소드를 가정해보자.
- -> 하나의 변수를 10개의 메소드에서 변경할 수 있다.
- 어떤 방법이 디버깅이 쉬울까?

중간 점검 문제

1. 자바에서 코드 재사용이 쉬운 이유는 관련된 __________와 ______________이 하나의 덩어리로 묶여 있기 때문이다.
2. 정보 은닉이란 __________을 외부로부터 보호하는 것이다.
3. 정보를 은닉하면 발생하는 장점은 무엇인가?

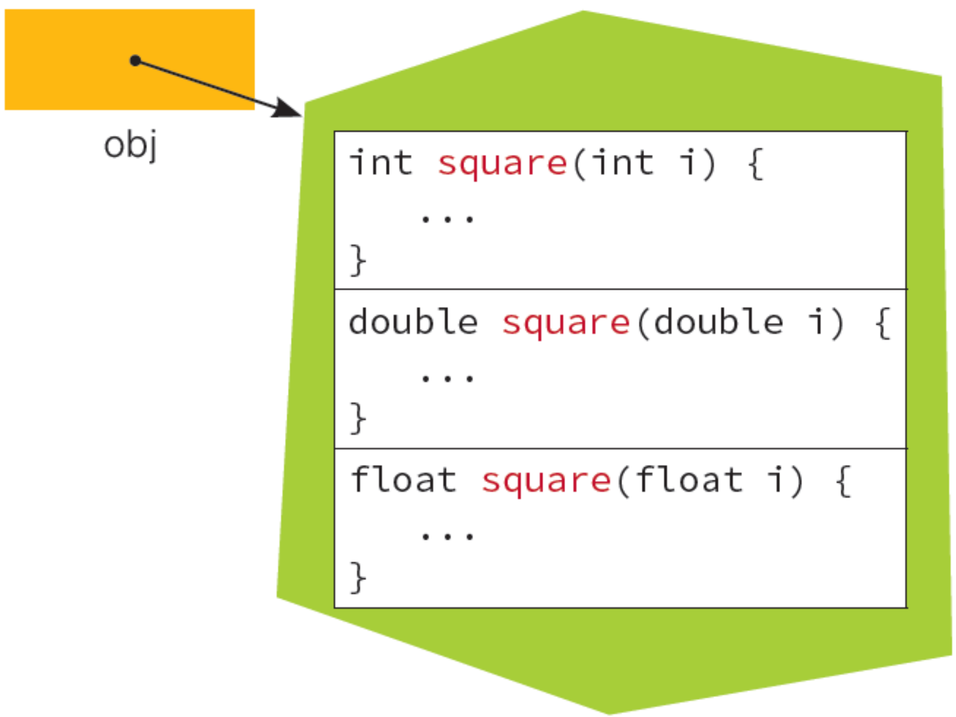
변수의 종류

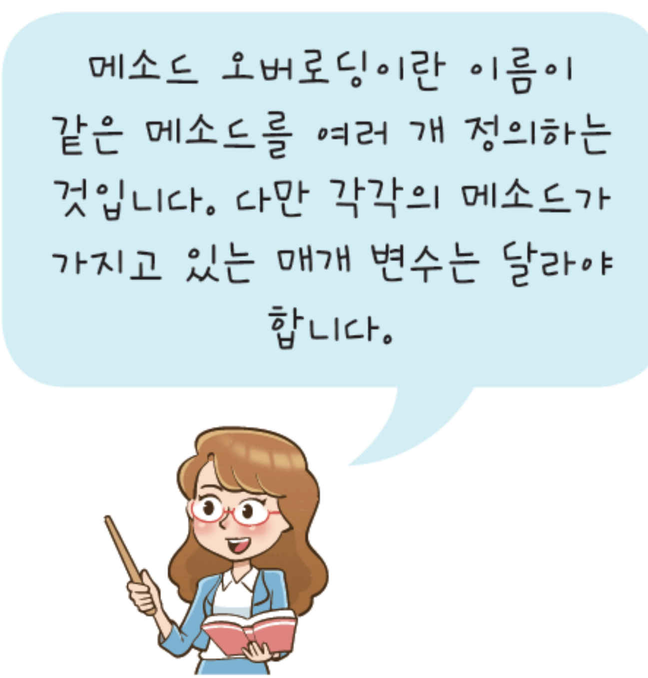
- 기초 변수(primitive variable)에는 실제 데이터값이 저장된다.
- 참조 변수(reference variable)는 참조 변수는 객체를 참조할 때 사용되는 변수로서 여기에는 객체의 참조값이 저장된다.



메소드 오버로딩

- 자바에서는 같은 이름의 메소드가 여러 개 존재할 수 있다. 이것을 **메소드 오버로딩(method overloading)**이라고 한다.





예제

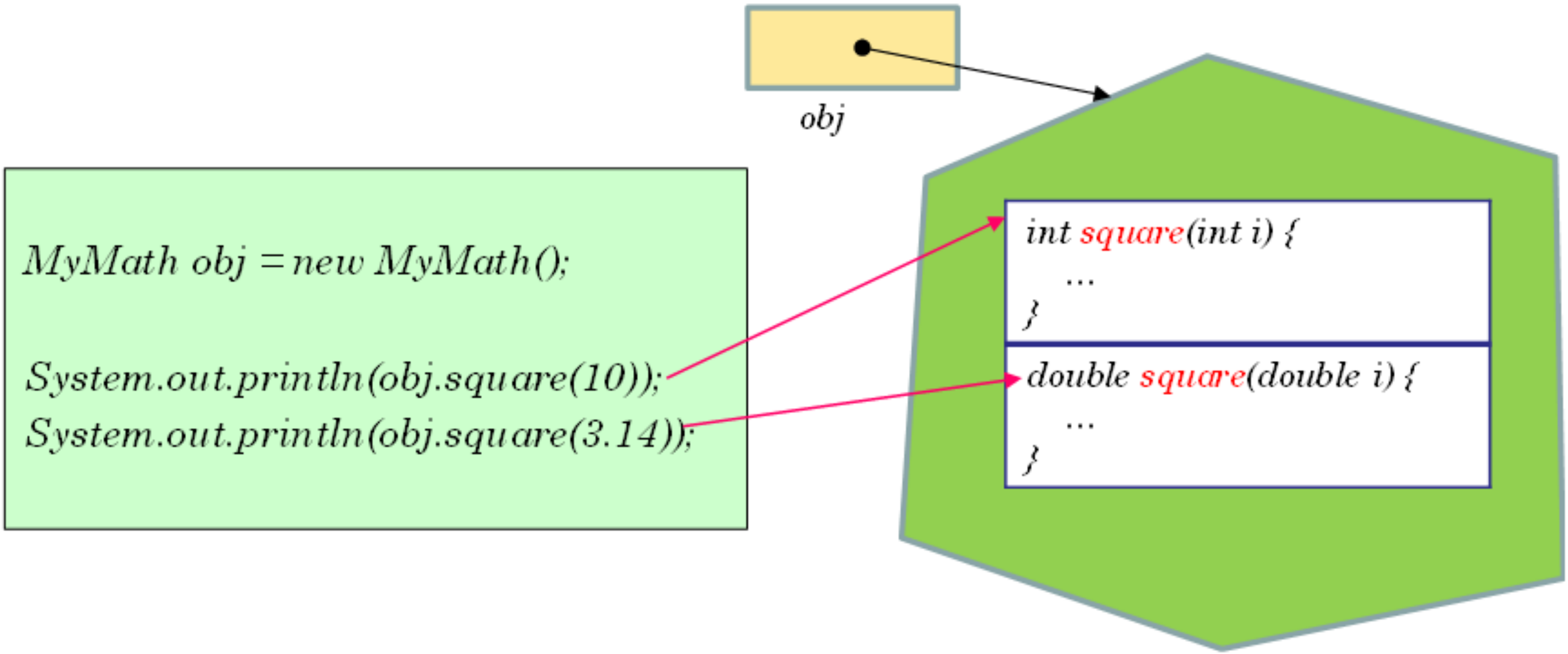
```
public class MyMath {
        // 정수값을 제곱하는 메소드
        int square(int i) {
                return i * i;
        }
        // 실수값을 제곱하는 메소드
        double square(double i) {
                return i * i;
        }
}
```

예제

```
public class MyMathTest {
        public static void main(String args[]) {
                MyMath obj = new MyMath();
                System.out.println(obj.square(10));
                System.out.println(obj.square(3.14));
        }
}
```


```
100
9.8596
```

예제 설명


obj
MyMath obj = new MyMath();
System.out.println(obj.square(10));
System.out.println(obj.square(3.14));
int square(int i) {
...
}
double square(double i) {
...
}

String 클래스

- 문자열은 자바에서 기초 자료형이 아니다.
- 그러나 문자열을 저장하고 처리하는 String이라고 하는 클래스가 존재한다

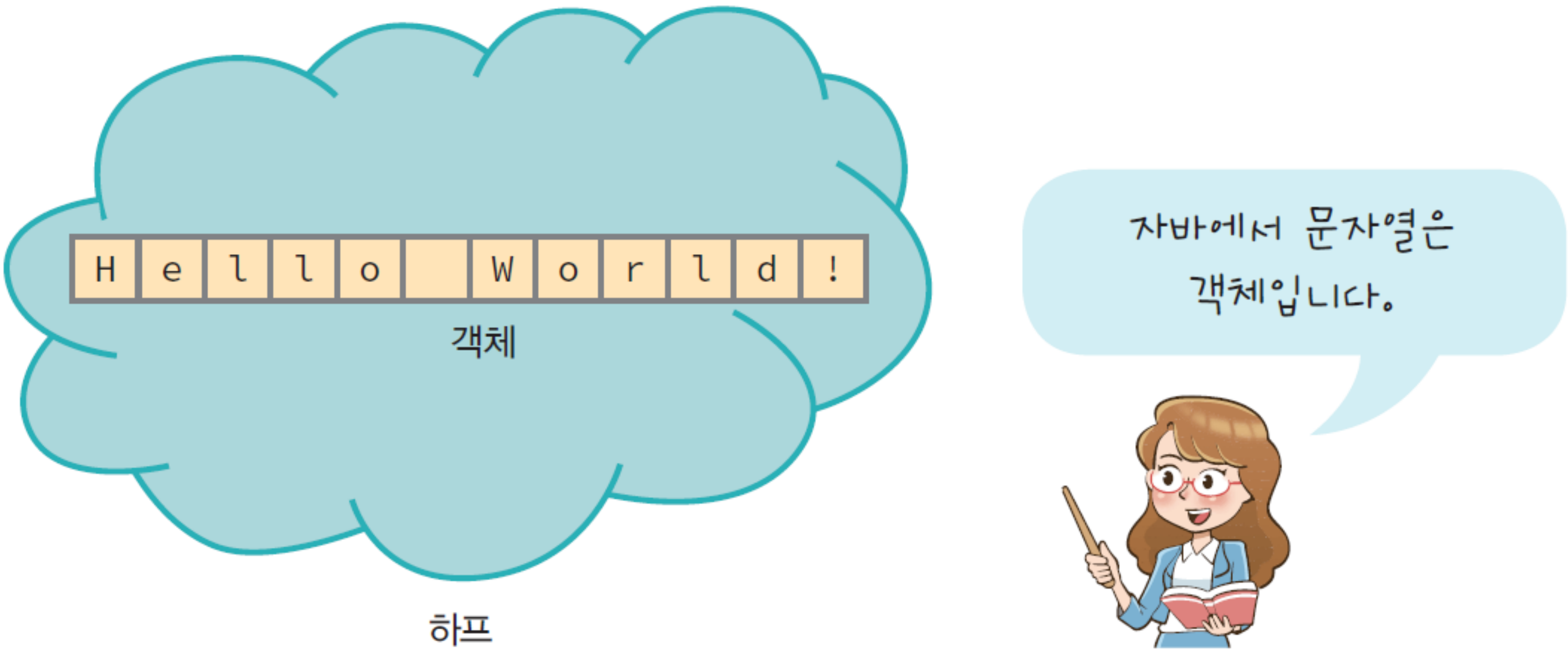
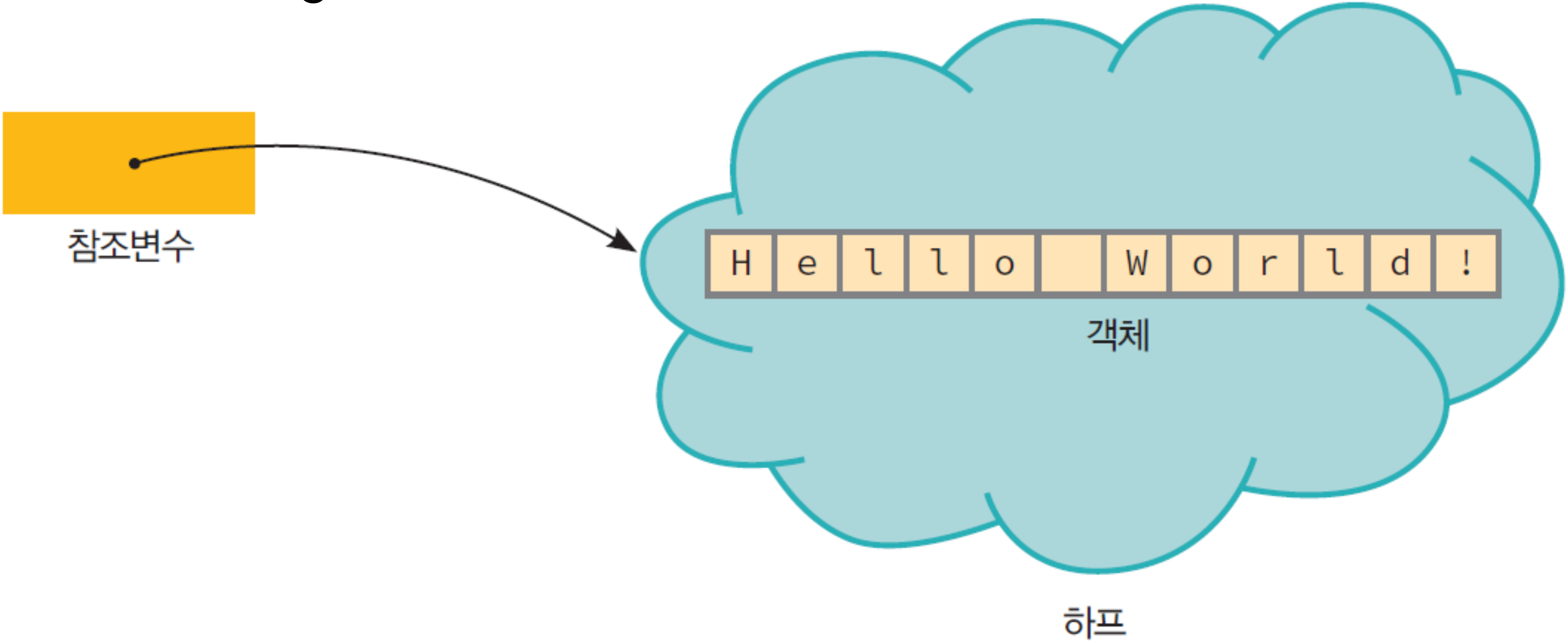


그림 5-2 • 문자열은 객체이다.

String 클래스의 객체 생성

- // 선언과 동시에 초기화
- String s = **new** String("Hello World!");
- String s = "Hello World!" // 동일한 역할함



String 클래스의 메소드

반환형	메소드 요약
char	charAt(int index) 지정된 인덱스에 있는 문자를 반환한다.
int	compareTo(String anotherString) 사전적 순서로 문자열을 비교한다. 앞에 있으면 −1, 같으면 0, 뒤에 있으면 1이 반환된다.
String	concat(String str) 주어진 문자열을 현재의 문자열 뒤에 붙인다.
boolean	equals(Object anObject) 주어진 객체와 현재의 문자열을 비교한다.
boolean	equalsIgnoreCase(String anotherString) 대소문자를 무시하고 비교한다.
boolean	isEmpty() length()가 0이면 true를 반환한다.
int	length() 현재 문자열의 길이를 반환한다.

String 클래스 사용하기

```
public class StringTest
{
    public static void main (String[] args)
    {
        String proverb = "A barking dog"; // new 연산자 생략
        String s1, s2, s3, s4;  // 참조 변수로서 메소드에서 반환된 참조값 받음

        System.out.println ("문자열의 길이 =" + proverb.length());

        s1 = proverb.concat (" never Bites!"); // 문자열 결합
        s2 = proverb.replace ('B', 'b');                // 문자 교환
        s3 = proverb.substring (2, 5);           // 부분 문자열 추출
        s4 = proverb.toUpperCase();              // 대문자로 변환

        System.out.println(s1);
        System.out.println(s2);
        System.out.println(s3);
        System.out.println(s4);
    }
}
```

예제


```
문자열의 길이 =13
A barking dog never Bites!
A barking dog
bar
A BARKING DOG
```

수치값-> 문자열

- 자바에서는 문자열과 기초 자료형 변수를 결합하게 되면 자동적으로 기초 자료형을 문자열로 변환한다.

```
int x = 20;
System.out.println("결과값은 " + x);
// "결과값은 20" 이 출력된다.
String answer = "The answer is " + 100;
// "The answer is 100"
```

문자열->수치값

- 즉 문자열 "123" 을 숫자 123으로 변환하려면 어떻게 하여야 하는가?
- 자바에는 이것을 전문으로 해주는 클래스가 있다. 바로 랩퍼 클래스인 Integer 클래스이다.

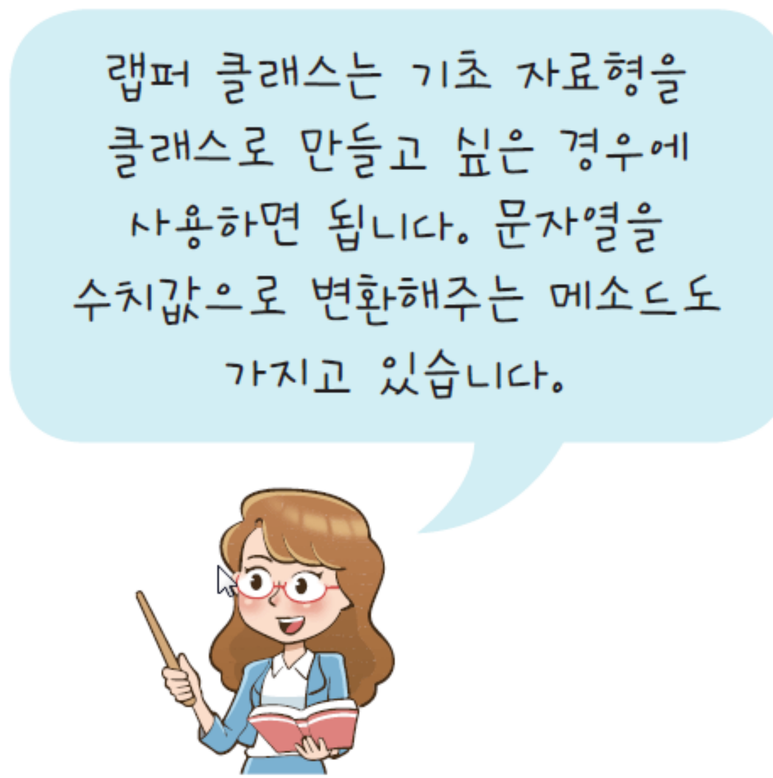
- 문자열을 기초 자료형으로 변환하려면 각 랩퍼 클래스의 parseXXX() 메소드를 사용한다.

```
int i = Integer.parseInt("123");
// 변수 i에 정수 123이 저장된다.
double d = Double.parseDouble("3.141592");
// 변수 d에 실수 3.141592가 저장된다.
```

문자열->수치값

- 자바에는 문자열을 수치값으로 변환해주는 클래스:

기초 자료형	랩퍼 클래스
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
char	Character
boolean	Boolean
void	Void



Q & A