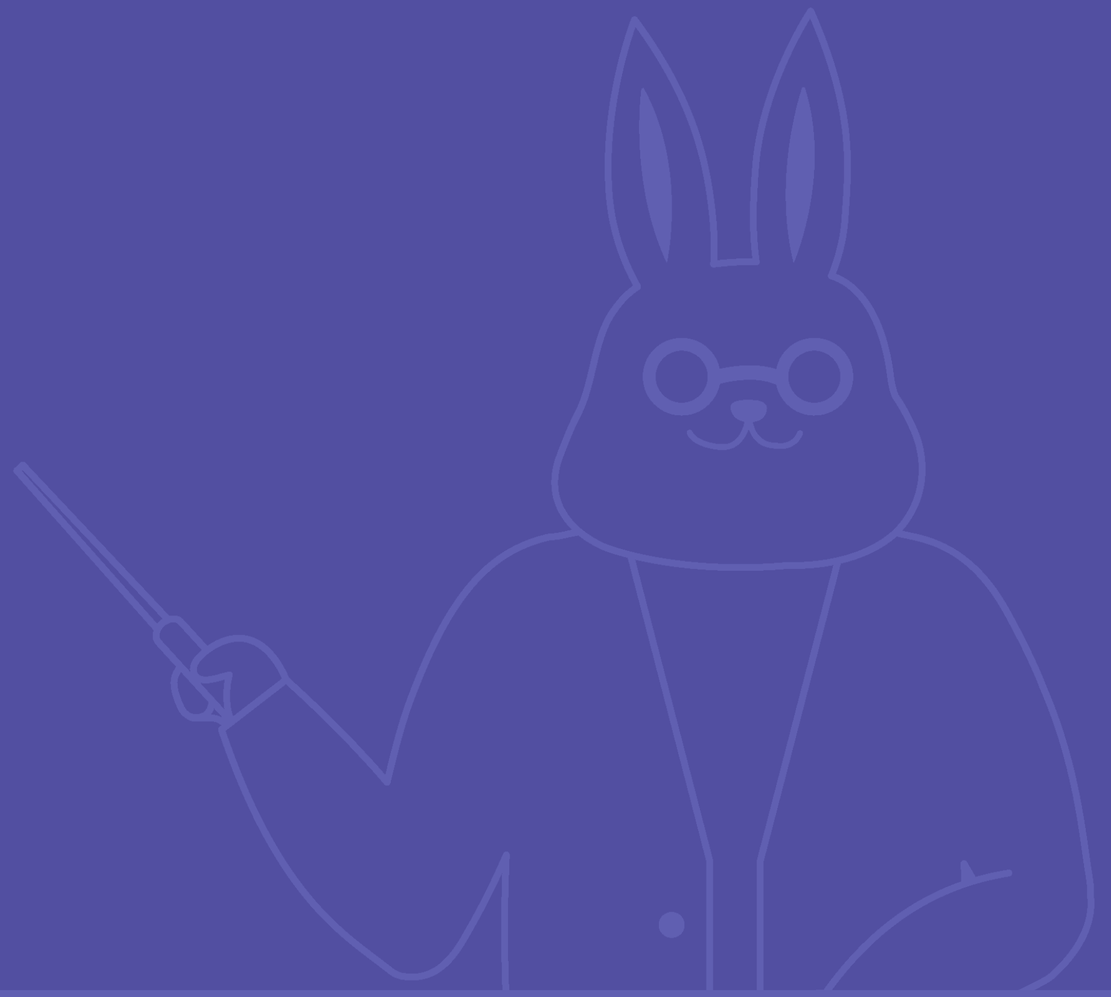
Java 2

4장 캡슐화와 정보 은닉



Contents

01

접근 제한자

접근 제한자

접근 제한자를 활용하여

클래스의 변수와 메소드를 보호

접근 제한자

접근 제한자	접근 범위
`public`	외부 클래스 어디서나
`protected`	상속 관계의 클래스 (부모 / 자식 관계)
`private`	같은 클래스 내부
`(default)`	같은 패키지 내부

캡슐화

객체지향 프로그래밍의 네 가지 특징

추상화, **캡슐화**, 상속성, 다형성

01 접근 제한자

캡슐화란?

사용자에게 **필요한 부분**만 공개하는 것

캡슐화란?

약을 먹는 사람은 **효과**가 중요하지

성분은 중요하지 않다.

캡슐화란?

자동차 운전자는 엑셀을 밟으면 **가속**이 된다는 것이 중요하지

엔진이 정확히 어떻게 **동작**하는지 아는 것은 중요하지 않다.

캡슐화란?

클래스 사용자에게 **필요한 부분**만 보여주는 것

캡슐화란?

클래스 사용자는
클래스가 어떻게 구현되었는지 알 필요가 없다.

내부 로직은 숨긴다.

접근 제한자

private는 클래스 밖에서의 접근을 막는다.

Example

```
class Student {
    private int number;     //학번
    int score;              //시험 점수
    String name;            //이름
}
...
Student s = new Student();
s.number = 1001; //Error. private 변수는 접근 불가
s.score = 99; //OK!
s.name = “Elice”; //OK!
```

/* elice */

접근 제한자

private 변수를 외부에서 사용하려면?

접근 제한자

private 변수에 접근하는

public 메소드를 제공해야 한다.

접근 제한자

get, set 메소드를 직접 제공한다.

Example

```
class Student {
    private int number;     //학번

    public int getNumber(){
        return number;
    }

    public void setNumber(int number){
        this.number = number;
    }
```

01 접근 제한자

접근 제한자를 사용하는 이유

private로 선언하고 public으로 접근할거면

그냥 public 변수를 직접 쓰면 되지 않나요?

접근 제한자를 사용하는 이유

잘못된 값 설정을 막을 수 있다.

Example

```
class Student {
    private int number;    //학번
    ...
    public void setNumber(int number){
        if (number < 1000 || number > 9999) {
            System.out.println(“올바른 학번이 아닙니다.”);
            return;
        }
        this.number = number;
    }
}
```

01 접근 제한자

접근 제한자를 사용하는 이유

클래스 내부에서만 사용할 변수나 메소드는

private로 선언

접근 제한자가 없는 경우

접근 제한자가 **없는** 경우는 무엇인가요?

01 접근 제한자

접근 제한자가 없는 경우

default 접근 제한이며

같은 패키지 내부에서만 접근 가능

public class

public 키워드가 **class** 앞에 오는 경우에는

어떤 의미인가요?

public class

class는 기본적으로 **default** 속성

01 접근 제한자

public class

public class는 해당 자바 파일의 **대표 클래스**이며
파일명과 클래스 이름이 일치해야 한다.

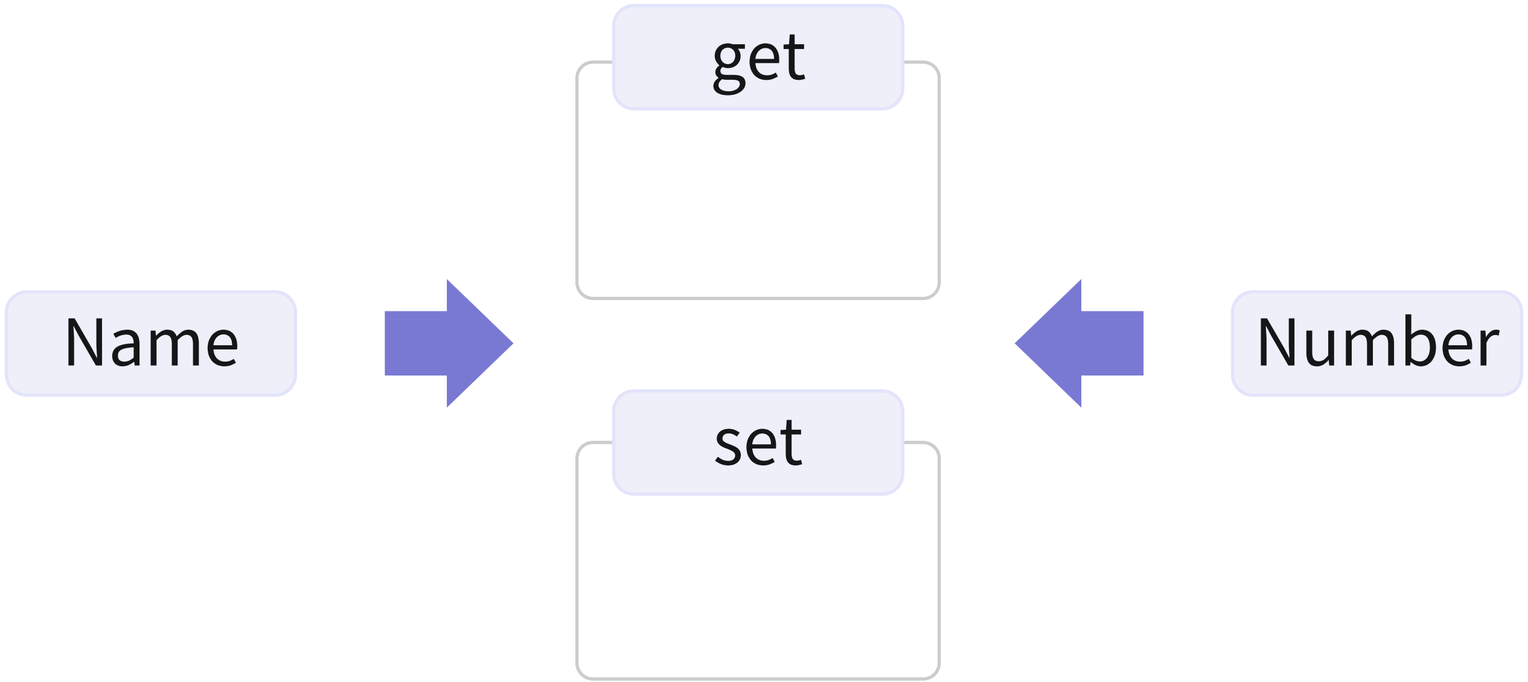
Example

```
public class Example {  //파일명이 Example.java
    public static void main(String[] args) {

    }
}
```

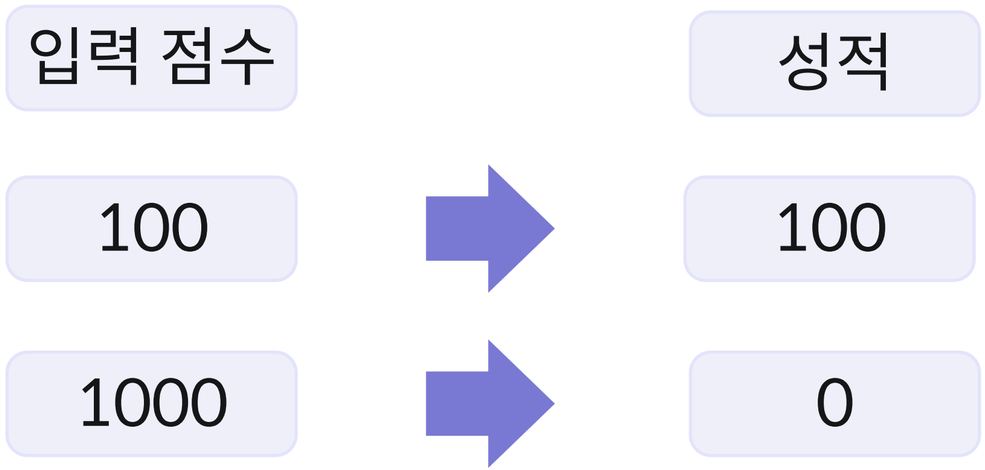
[실습1] 캡슐화 활용해보기(1)

get, set메소드를 구현하여 정보를 가져와봅시다!



[실습2] 캡슐화 활용해보기(2)

성적 조작을 캡슐화를 통해 막아봅시다!



02

클래스의 상호작용

객체지향 프로그래밍

클래스를 속성과 기능으로 설계하고

객체들의 **상호작용**으로 프로그램을 작성하는 것

자바 프로그래밍

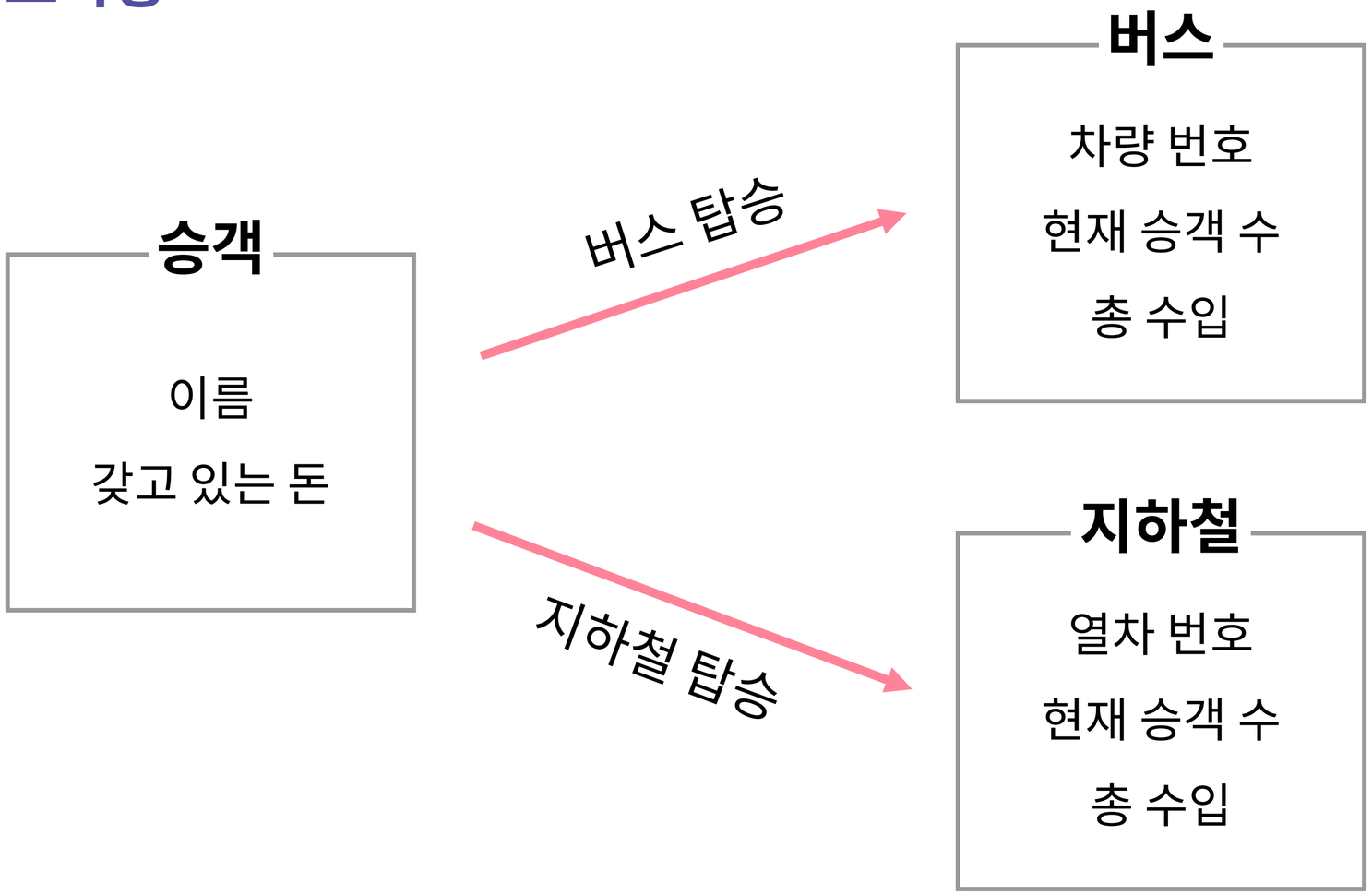
자바는 객체지향 프로그래밍 언어이며

모든 것은 **클래스** 단위로 이루어진다.

클래스의 상호작용

승객이 버스와 지하철을 이용하는 과정을 작성하려면?

클래스의 상호작용



클래스의 상호작용

지금까지 배운 내용으로

클래스의 상호작용을 구현해봅시다.

[실습3] 대중교통 이용하기

버스에 탑승한 승객과 누적 수입을 출력해봅시다!

03

Singleton Pattern

Singleton Pattern(싱글턴 패턴)이란?

프로그램 전반에 이용될 인스턴스를

단 하나만 생성하는 디자인 패턴

Singleton Pattern 만들기

교내 학생들을 관리하는 프로그램

학생은 여러 명이지만, 학교는 **단 하나**

03 Singleton Pattern

Singleton Pattern 만들기

1. private 생성자 만들기

Example

```
class School {
    //외부에서의 객체 생성을 금지함
    private School() {};
}
```

/* elice */

Singleton Pattern 만들기

2. 클래스 안에 **static 인스턴스** 만들기

Example

```
class School {
    //프로그램 전체에서 사용할 유일한 인스턴스
    private static School instance;
}
```

Singleton Pattern 만들기

3. 객체를 얻을 수 있는 **public 메소드 제공**

Example

```
class School {
    //외부에서의 객체 생성을 금지함
    public static School getInstance(){
        if(instance == null)
            instance = new School();
        return instance;
    }
}
```

/* elice */

Singleton Pattern 만들기

4. 싱글턴 객체 사용

Example

```
...
School mySchool = School.getInstance();

mySchool.addStudent(“Elice”);
...
```

/* elice */

Singleton Pattern의 사용

디자인 패턴은 상황에 맞는 적용법일 뿐
고유한 장/단점은 없다.

인스턴스가 프로그램에 단 1개만 존재해야 하는 경우에 적용

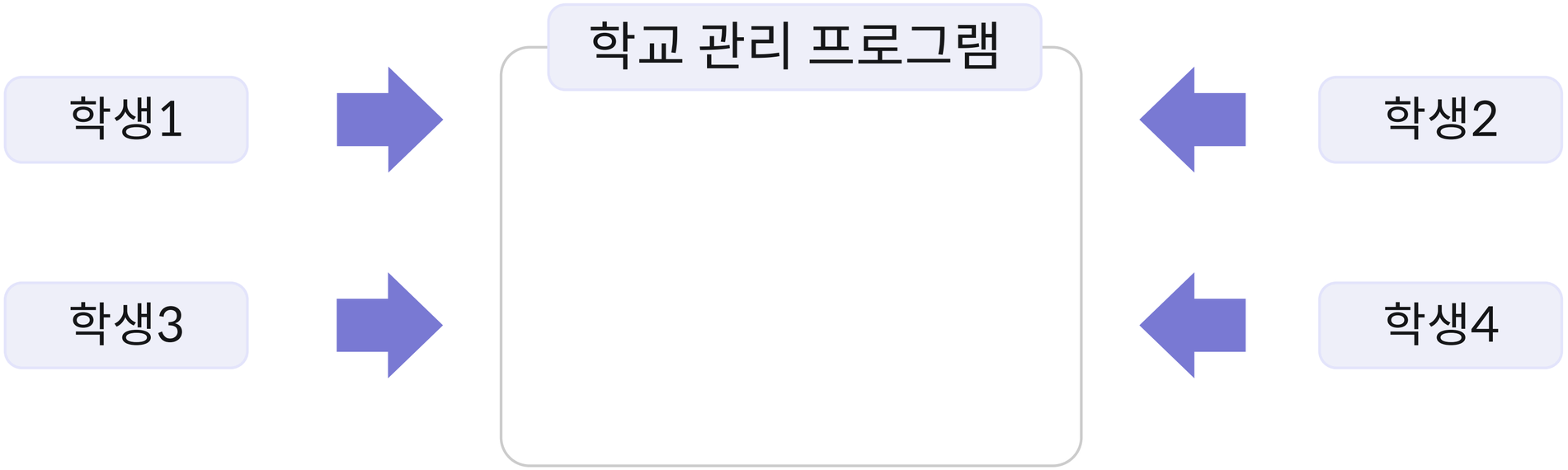
Singleton Pattern의 장점

static 인스턴스이기 때문에

데이터 공유에 용이

[실습4] 교내 학생 관리하기

하나의 인스턴스로 학교 인원을 관리해봅시다!



Credit

/* elice */

코스 매니저
강윤수

콘텐츠 제작자
강윤수

강사
유동환 선생님

디자인
박주연

Contact

TEL
070-4633-2015

WEB
https://elice.io

E-MAIL
contact@elice.io