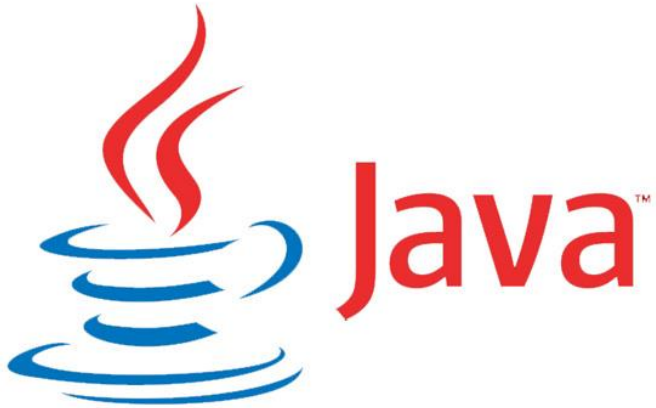


5장 객체지향 프로그래밍

김철학

목차

1. 객체지향 프로그래밍 개요
2. 클래스와 객체
3. 캡슐화
4. 클래스 변수, 메서드
5. 상속
6. 메서드 Overriding
7. 다형성
8. 추상클래스
9. 인터페이스

1. 객체지향 프로그래밍 개요

- 객체지향 프로그래밍Object-Oriented Programming 은 객체를 바탕으로 객체 간의 상호 작용으로 프로그램을 개발하는 프로그래밍 기법
- 객체Object 는 현실세계의 유무형의 하나의 독립적인 개체로 속성과 기능으로 구성
- 객체지향 프로그래밍은 코드의 재사용성, 확장성, 유지보수성을 높여 소프트웨어 품질 향상에 기여

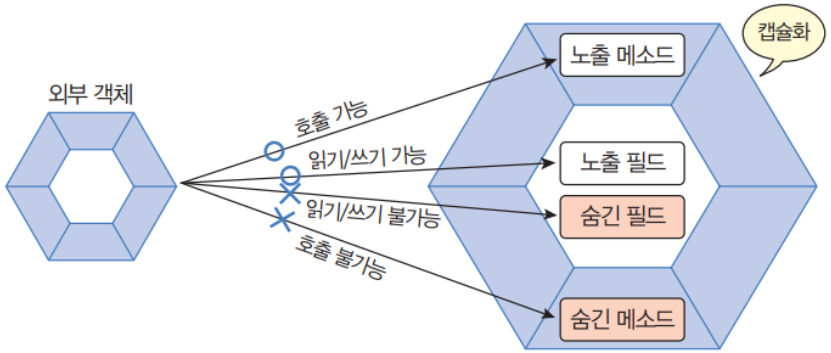
주요 특징	내용
추상화Abstraction	객체의 공통적인 속성과 기능을 추출하여 클래스로 정의하는 특징
캡슐화Encapsulation	객체의 정보 노출을 최소화하고 꼭 필요한 정보만 제공하는 특징
상속Inheritance	기존에 만들어 놓은 클래스의 확장 시켜 기능을 재사용하는 특징
다형성Polymorphism	하나의 객체가 여러 가지 타입으로 변환을 통해 다양한 기능을 제공하는 특징

2. 객체와 클래스

- 클래스Class 는 객체를 생성하는 틀이며, 속성과 메서드로 구성
- 현실세계의 객체를 추상화 시킨 대상이 클래스이며, 클래스를 통해 객체를 생성
- 인스턴스Instance 는 클래스로부터 생성된 메모리상의 객체

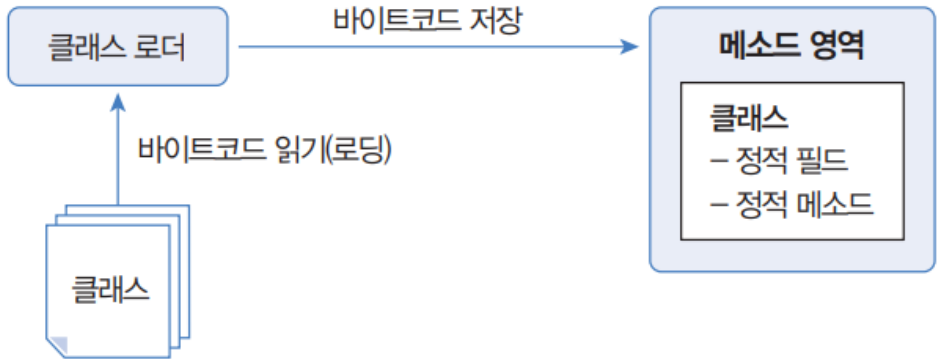
3. 캡슐화

- 캡슐화는 객체의 내용을 외부에서 알지 못하도록 객체의 정보를 은닉하는 특성
- 접근 제한자를 사용해 캡슐화 구현
- 은닉된 정보의 안전한 제공을 위해 Getter와 Setter를 제공



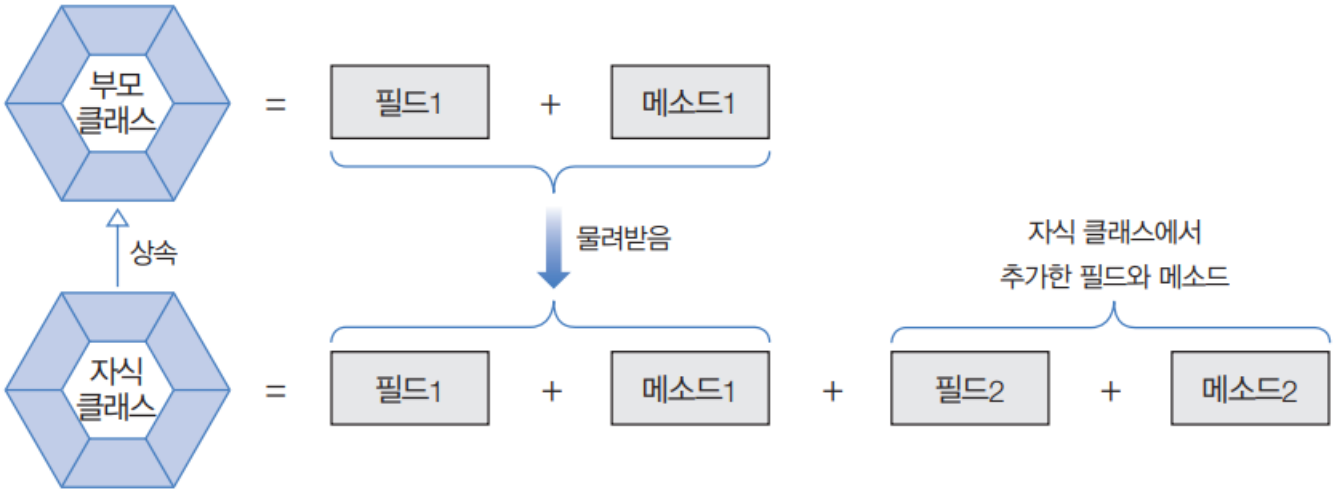
4. 클래스 변수, 메서드

- Java는 물리적인 메모리 공간을 논리적인 Stack, Heap, Method Area로 나누어 사용
- 클래스 변수, 클래스 메서드는 static 선언한 정적타입 특성으로 객체 생성 없이 사용
- 싱글톤 객체는 static 선언 객체로 메서드 영역에 오직 하나의 인스턴스로 존재



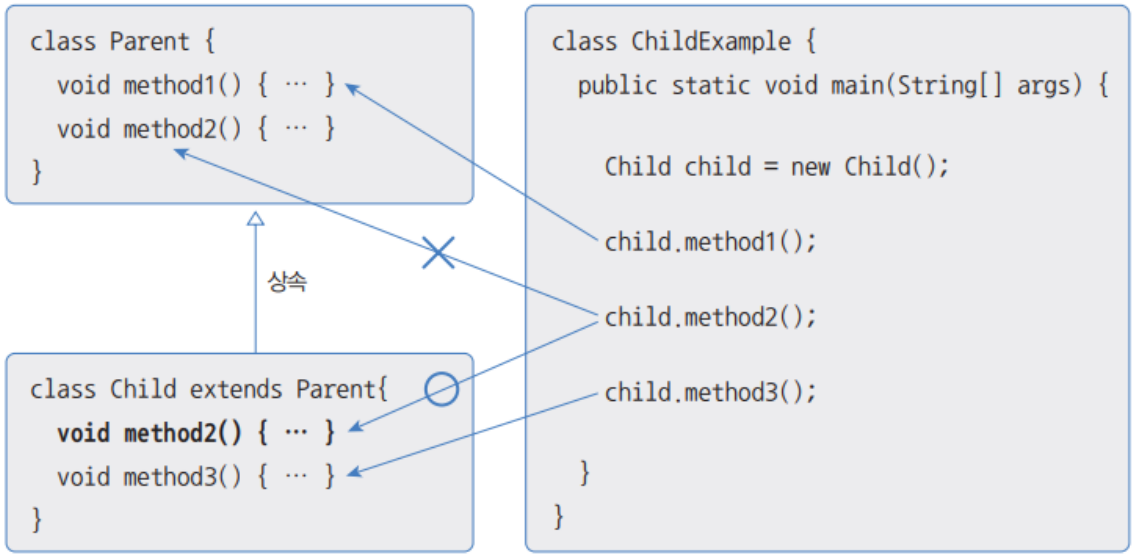
5. 상속

- 상속은 기존의 클래스가 가지고 있는 필드와 메서드를 그대로 물려 받은 자식 클래스를 정의
- 공통적인 내용을 부모 클래스에 두고 자식 클래스에서 상속 받아 일관되고 효율적인 프로그래밍 수행



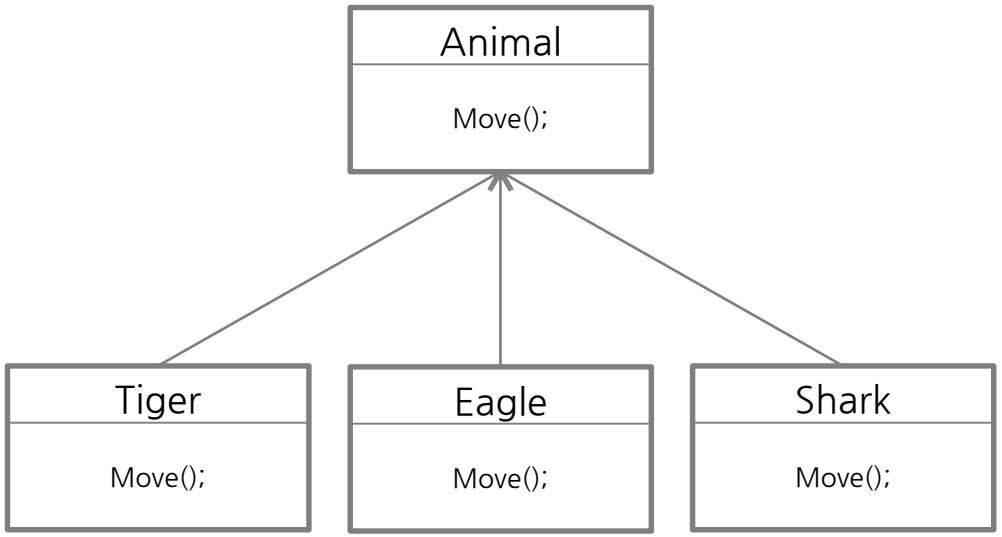
6. 메서드 Overriding

- 메서드 재정의Override 는 부모 클래스의 기능을 자식 클래스에 맞게 다시 정의하는 문법
- 부모 클래스의 기능을 자식 클래스에서 다시 재정의해서 다형성 구현



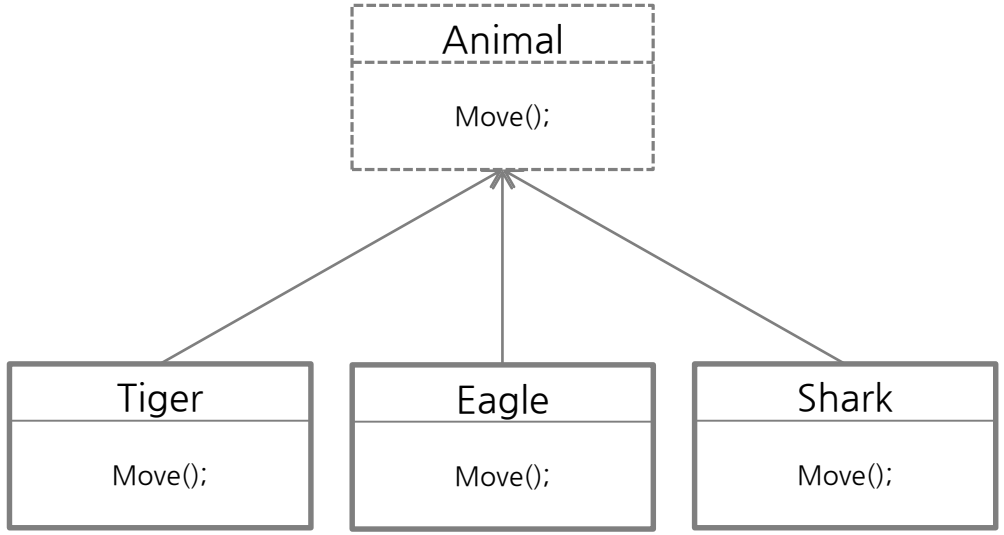
7. 다형성

- 다형성Polymorphism 은 상속과 메서드 재정의로 부모클래스의 기능이 자식 클래스에서 여러 기능으로 변할 수 있는 성질
- 객체지향 프로그래밍에서는 다형성을 통해 코드의 반복을 줄이고 유연성을 향상



8. 추상 클래스

- 추상Abstract 클래스는 객체지향 개념을 실제 프로그램 개발에 쉽게 적용하고 유연한 설계를 위한 클래스
- 추상 클래스는 클래스를 정의하기 위한 프로토타입으로 실제 클래스 구현은 자식 클래스에서 담당



9. 인터페이스

- 인터페이스Interface 는 클래스간 공통의 통일된 표준 구조를 설계하기 위한 문법 요소
- 인터페이스는 오직 추상 메서드만 포함하며, 일반적으로 다형성 구현에 인터페이스를 사용