03-08 [Java]

소유자	종수 김
태그	

[접근 제어자]

해당 클래스 외부에서 특정 필드나, 메서드에 접근하는 것을 허용하거나 제한.
즉, 속성과 기능을 외부로부터 숨기는 것.

특정 메서드를 통해 요구사항을 구현하더라도, 멤버 변수에 직접 접근해서 값을 바꾼다면 요구사항의 부재가 생길 수 있음.

[접근 제어자 종류]

private

해당 접근 제어자가 붙은 변수 혹은 메서드는, 본인 클래스 내부에서만 사용이 가능하고 외부에서 접근을 막음.

default

같은 패키지 내부에서만 호출 허용

접근 제어자를 명시하지 않으면 자동으로 default로 허용

package-private: 접근 제어자를 사용하는 멤버는 동일한 패키지 내의 다른 클래스에서만 접근이 가능하기 때문.

protected

같은 패키지 안에서 호출은 허용하나, 패키지가 달라도 상속 관계의 호출만 허용

즉, 패키지 + 상속

public

모든 외부 호출을 허용

순서대로 많이 허용

- private < default < protected < public
- 지역 변수내에는 사용 불가능
- 클래스, 멤버 변수, 메서드에만 사용 가능.

- 생성자도 접근 제어자 관점에서는 메서드와 같다.

[접근 제어자 사용 - 클래스 레벨]

- 클래스 레벨의 접근 제어자는 public, default만 사용할 수 있다.
- public class는 파일명과 클래스명이 무조건 같아야만 한다.
  - 하나의 java파일의 public 클래스는 하나만 가능하다.
  - 하나의 자바 파일에 default 접근 제어자는 무한히 만들 수 있다.

[캡슐화]

객체 지향 프로그래밍의 중요한 개념중 하나.
데이터와 해당 데이터를 처리하는 메서드를 하나로 묶어서 외부에서의 접근을 제한하는 것.

- 속성과 기능을 하나로 묶고, 외부에 꼭 필요한 기능만 노출하고 모두 내부로 숨기는 것.

캡슐화를 안전하게 완성할 수 있게 해주는 장치가 바로 접근 제어자.

1. 데이터를 숨겨라

   객체에는 속성(데이터)과 기능(메서드)이 있다.
   캡슐화에서 가장 필수로 숨겨야 하는 것은 속성(데이터).

   객체 내부의 데이터를 외부에서 함부로 접근하게 두면, 클래스 안에서 데이터를 다루는 모든 로직을 다 무시하고 데이터를 변경할 수 있기 때문.

   운전할 때를 떠올려보면, 우리는 속도를 높이기 위해 자동차 부품을 다 열어서 속도계를 조절하지 않음.
   단순, 엑셀(기능)을 밟으면 속도가 올라가고, 자동차가 빠르게 가는 기능을 사용하는 것.

   객체의 데이터는 객체가 제공하는 기능인 '메서드'를 통해서 접근해야만 함.

2. 기능을 숨겨라

   객체의 기능 중, 외부에서 사용하지 않고 내부에서만 사용하는 기능들 또한 숨기는 것이 좋다.

   자동차를 운전하기 위해 엑셀, 핸들 정도의 기능만 알면될뿐, 엔진 조절 기능, 배기 기능 등 기능을 사용하기 위해 정의한 복잡한 기능까지는 알 필요가 없는 것.

- 데이터는 모두 숨기고, 기능은 꼭 필요한 기능만 노출하는 것이 좋은 캡슐화.
- public으로 선언된 메서드, 필드는 다른 곳에서 사용해도 된다는 허락 표시.