Chapter 13-1

접근 제한자

접근 제한자란?

은행

접근 제한자(Access Modifier)

- 접근 제한자는 클래스와 멤버변수, 메서드, 생성자의 접근을 제어할 수 있는 제한자입니다.

- 라이브러리 클래스를 설계할 때는 외부 클래스에서 접근할 수 있는 멤버와 접근할 수 없는 멤버로 구분해서 변수, 생성자, 메서드를 설계하는 것이 바람직합니다.

- 외부에서 객체 생성을 막기 위해 생성자를 호출하지 못하게 하거나 객체의 특정 데이터를 보호하기 위해 해당 멤버변수에 접근하지 못하도록 막는 것이 접근 제한자의 역할입니다.

- 클래스에는 접근 제한자를 public과 default만 붙일 수 있습니다.

접근 제한자의 종류

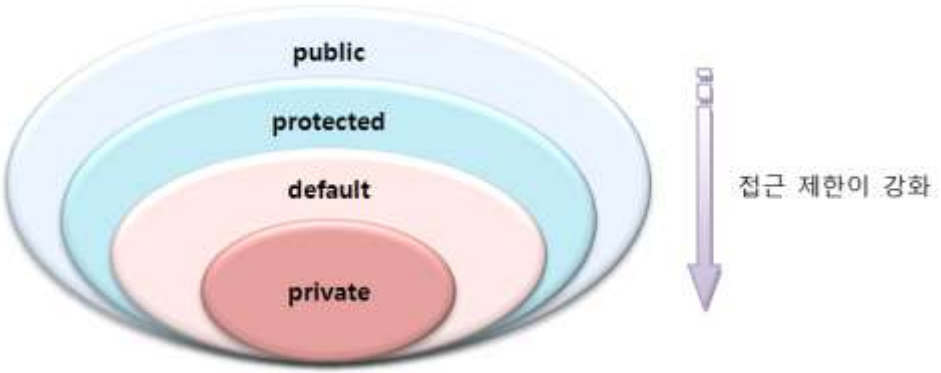
접근 제한자의 종류

1. public: 같은 클래스, 같은 패키지, 다른 패키지를 막론하고 접근이 가능.

2. protected: 같은 클래스, 같은 패키지는 접근이 가능하지만 다른 패키지에 속해있는 클래스인 경우 상속관계가 없으면 접근이 불가능.

3. default(접근 제한자를 붙이지 않는 형태): 같은 클래스, 같은 패키지에서만 접근이 가능하며 패키지가 다를 경우 접근이 불가능.

4. private: 같은 클래스 내부가 아니면 접근이 불가능.



접근 제한	적용할 내용	접근할 수 없는 클래스
public	클래스, 필드, 생성자, 메소드	없음
protected	필드, 생성자, 메소드	자식 클래스가 아닌 다른 패키지에 소속된 클래스
default	클래스, 필드, 생성자, 메소드	다른 패키지에 소속된 클래스
private	필드, 생성자, 메소드	모든 외부 클래스

Chapter 13-2

객체지향 프로그래밍 기술 (상속, 은닉, 다형성)

정보은닉-캡슐화

은닉(Encapsulation)캡슐화

- 은닉은 사용자에게 상세한 내부 구현을 숨기고 필요한 부분만 보이게 하는 것입니다.

- 은닉을 사용하기 위해서는 클래스의 멤버변수의 접근제한자를 private으로 설정합니다.

- 은닉된 멤버변수에 접근하기 위해서는 공개된(public) 메서드를 통해서 접근할 수 있는데, 변수의 값을 변경시키는데 사용되는 메서드는 setter메서드라고 부르며, 변수의 값을 얻어오는데 사용하는 메서드를 getter메서드라고 부릅니다.

- 이처럼 공개 메서드를 이용하여 데이터를 변경시킬 경우 메서드 내에 데이터 유효성을 검증할 수 있는 루틴을 넣을 수 있습니다.

- 뿐만 아니라 경우에 따라 접근 권한을 체크할 수 있는 로직을 포함시키면 인가되지 않은 사용자에게 중요한 데이터나 로직을 숨길 수도 있습니다.

- 멤버변수만 private 제한자를 가지는 것은 아닙니다. 외부에 공개하고 싶지 않은 메서드들도 private으로 선언할 수 있습니다.

은닉 - badcase

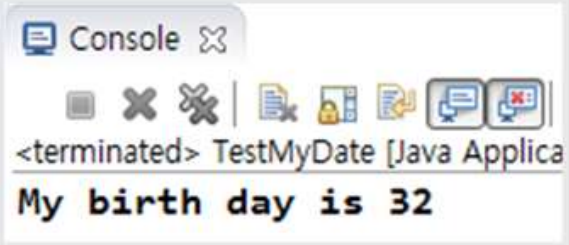
encapsulation/badcase/MyDate.java

```
1:  package encapsulation.badcase;
2:
3:  public class MyDate {
4:     public int day;
5:     public int month;
6:     public int year;
7:  }
```

변수의 접근제한을 열어두면 모든 값이 저장 될 수 있다.
하지만, 잘못된 값이 저장될 수도 있다는 뜻이다

encapsulation/badcase/TestMyDate.java

```
1:  package encapsulation.badcase;
2:
3:  public class TestMyDate {
4:     public static void main(String[] args)   {
5:        MyDate myBirth = new MyDate();
6:        myBirth.day = 32;
7:        System.out.println("My birth day is " + myBirth.day);
8:     }
9:  }
```



은닉-goodcase

encapsulation/goodcase/MyDate.java

```
1:  package encapsulation.goodcase;
2:
3:  public class MyDate {
4:     private int day;
5:     private int month;
6:     private int year;
7:     public void setDay(int d) {
8:        if( (d<1) || (d>31) ) {
9:           System.out.println("잘못된 날짜입력입니다");
10:          day = 1;
11:       } else {
12:          day = d;
13:       }
14:    }
15:    public int getDay() {
16:       return day;
17:    }
18:
19:    //month와 year필드의 set/g
20:
21: }
```

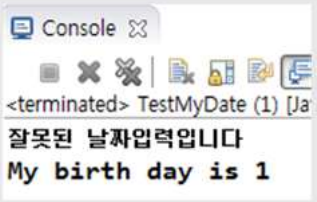
private 제한자는 외부에 접근이 불가능하다

저장하는 메서드 setter
값을 조회하는 메서드 getter의 사용을 강제화 한다

getter, setter의 사용시 검증하는 로직을 메서드에 기술 할 수 있다.

encapsulation/goodcase/TestMyDate.java

```
1:  package encapsulation.goodcase;
2:
3:  public class TestMyDate {
4:     public static void main(String[] args)   {
5:        MyDate myBirth = new MyDate();
6:        myBirth.day = 32; //Error
7:        myBirth.setDay(32);
8:        System.out.println("My birth day is " + myBirth.getDay());
9:     }
10: }
```



getter, setter메서드의 의미를 반드시 정리하자

Chapter 13

수고하셨습니다