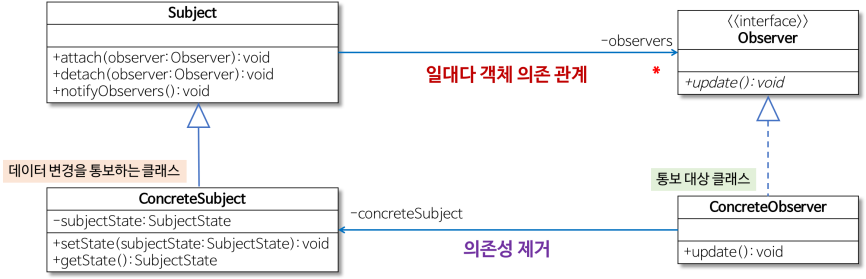
옵저버 패턴

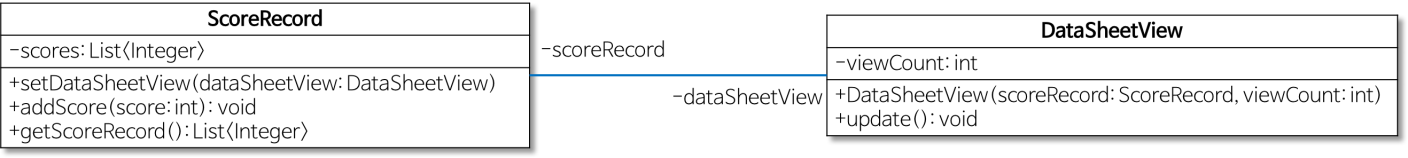
* 한 객체의 상태변화에 따라 다른 객체의 상태도 연동되도록 일대다 객체 의존 관계를 구성하는 패턴
  + 데이터의 변경이 발생했을 경우 상대 클래스나 객체에 의존하지 않으면서 데이터 변경을 통보하고자 할 때 유용하다.
    - Ex)새로운 파일이 추가되거나 기존 파일이 삭제되었을 때 탐색기는 다른 탐색기에게 즉시 변경을 통보해야 한다.
    - Ex)차량 연료량 클래스는 연료량이 부족한 경우 연료량에 관심을 가지는 구체적인 클래스(연료량 부족 경고 클래스, 주행 가능 거리 출력 클래스) 에 직접 의존하지 않는 방식으로 연료량 변화를 통보해야 한다.
  + 행위(Behavioral) 패턴의 하나



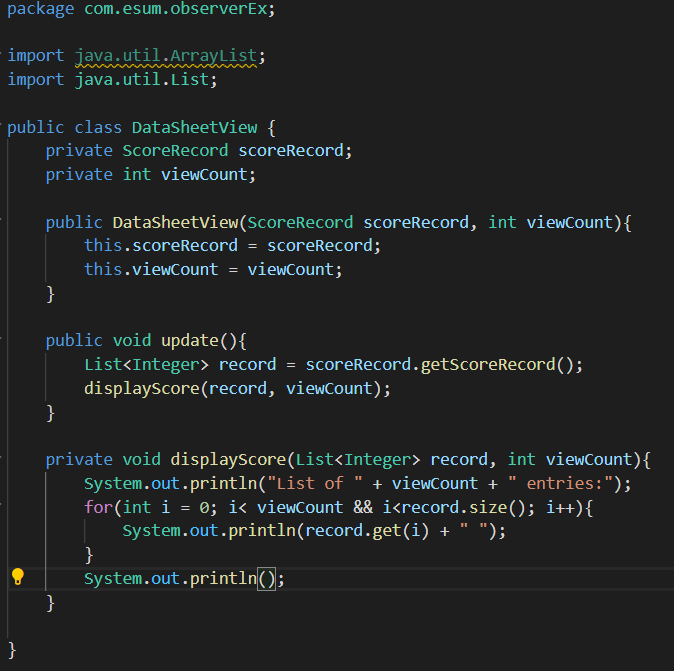
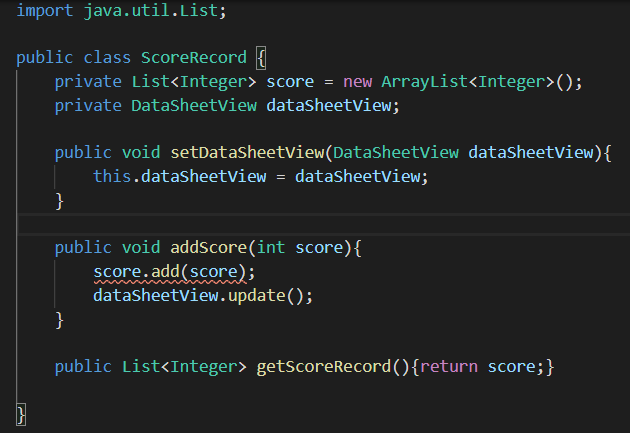
* + 옵저버 패턴은 통보 대상 객체의 관리를 Subject 클래스와 Observer인터페이스로 일반화
    - 이를 통해 데이터 변경을 통보하는 클래스(ConcreteSubject)는 통보 대상 클래스나 객체(ConcreateObserver)에 대한 의존성을 없앨 수 있다.
    - 결과적으로 통보 대상 클래스나 대상 객체의 변경에도 통보하는 클래스(ConcreteSubject)를 수정 없이 그대로 사용할 수 있다.
  + 역할이 수행하는 작업
    - Observer
      * 데이터의 변경을 통보받는 인터페이스
      * 즉, subject에서는 Observer 인터페이스의 update 메서드를 호출함으로써 ConcreteSubject의 데이터 변경을 ConcreteObserver에게 통보한다
    - Subject
      * ConcreteObserver 객체를 관리하는 요소
      * Observer 인터페이스를 참조해서 ConcreteObserver를 관리하므로 ConcreteObserver의 변화에 독립적일 수 있다.
    - ConcreteSubject
      * 변경 관리 대상이 되는 데이터가 있는 클래스(통보하는 클래스)
      * 데이터 변경을 위한 메서드인 setState가 있다.
      * setState 메서드에서는 자신의 데이터인 subjectState를 변경하고 Subject의 notifyObservers메서드를 호출해서 ConcreteObserver 객체에 변경을 통보한다.
    - ConcreteObserver
      * ConcreteSubject의 변경을 통보받는 클래스
      * Observer 인터페이스의 update메서드를 구현함으로써 변경을 통보받는다.
      * 변경된 데이터는 ConcreteSubject getState 메서드를 호출함으로써 변경을 조회한다.
  + 참고
    - 행위 패턴
      * 객체나 클래스 사이의 알고리즘이나 책임 분배에 관련된 객체
      * 한 객체가 혼자 수행할 수 없는 작업을 여러 개의 객체로 어떻게 분배하는지, 또 그렇게 하면서도 객체 사이의 결합도를 최소화하는 것에 중점을 둔다.

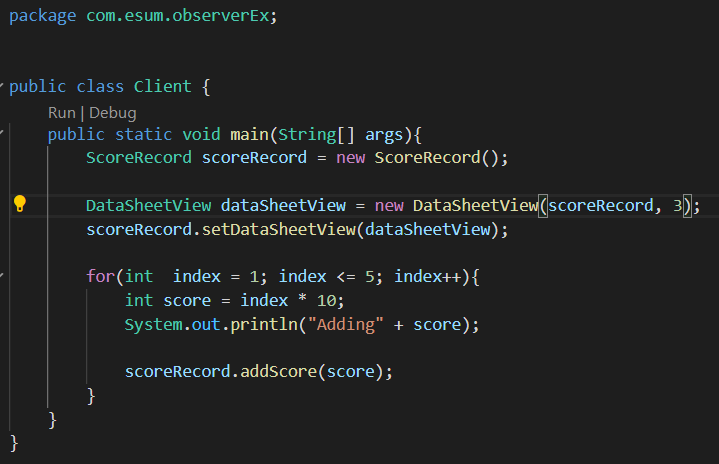
예시

여러가지 방식으로 성적 출력하기

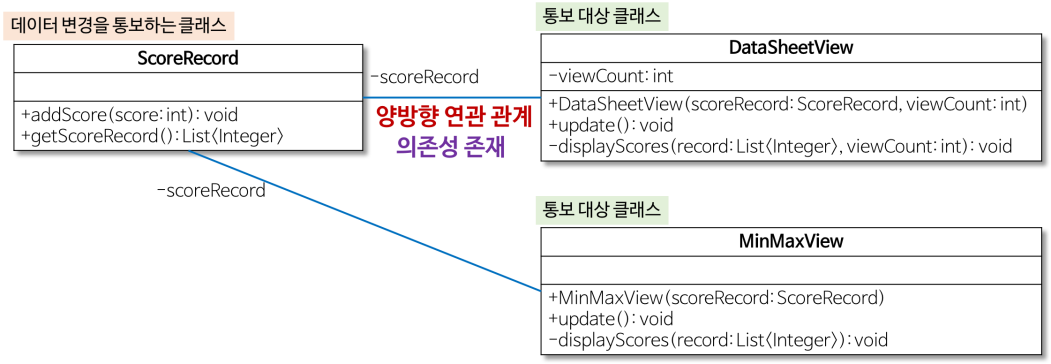


* + 입력된 성적 값을 출력하는 프로그램

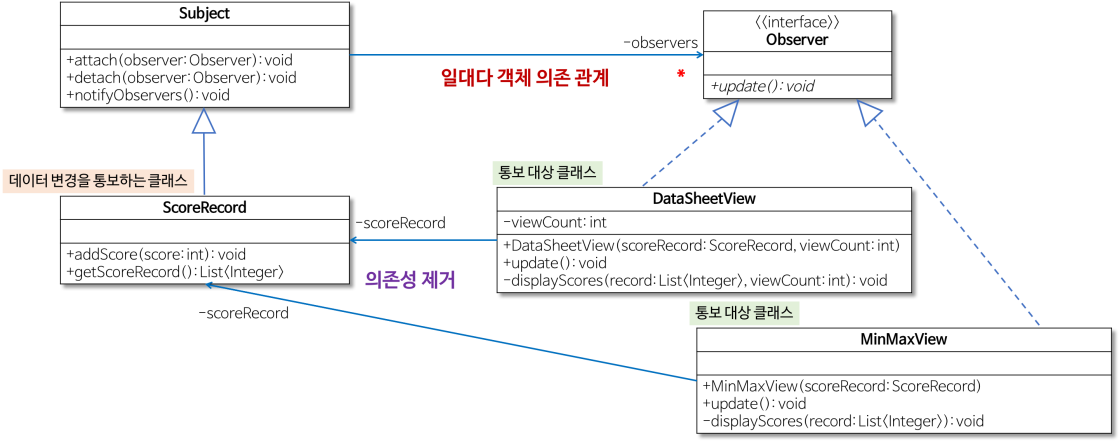




* + - ScoreRecord 클래스의 addScore메서드가 호출되면 ScoreRecord클래스는 자신의 필드인score객체에 점수를 추가한다.
      * 그리고 DataSheetView 클래스의 Update메서드를 호출함으로써 성적을 출력하도록 요청한다.
    - DataSheetView 클래스는 ScoreRecord 클래스의 getScoreRecord 메서드를 호출해 출력할 점수를 구한다.
      * 이때 DataSheetView 클래스의 update 메서드에는 구한 점수 중에서 명시된 개수만큼의 점수만을 출력한다.
* 문제점
  + 성적을 다른 형태로 출력하는 경우
    - 성적을 목록으로 출력하지 않고 성적의 최소/최대 값만 출력하려면
    - 정수 변경에 대한 통보 대상 클래스가 다른 대상 클래스로 바뀌면 기존 코드(ScoreRecord)의 내용을 수정해야하므로 ocp에 위배
  + 동시 혹은 순차적으로 성적을 출력하는 경우
    - 성적이 입력되었을 떄 최대 3개 목록, 최대 5개 목록, 최소/최대 값을 동시에 출력하려면?
    - 처음에는 목록으로 출력하고 나중에는 최소/최대 값을 출력하려면?
    - 이 경우에도 정수 변경에 대한 통보 대상 클래스가 다른 대상 클래스(DataSheetView -> MinMaxView)로 바뀌면 기존 코드(ScoreRecord)의 내용을 수정해야 하므로 OCP에 위배된다.
    - 즉 성적 변경을 새로운 클래스에 통보할 때마다 ScoreRecord클래스이 코드를 수정해야 하므로 재사용하기가 어렵다.



* + 이를 통해 성적 통보 대상이 변경되더라도 ScoreRecord클래스를 그대로 재사용할 수 있다.
  + ScoreRecord클래스에서 하는 작업
    - 통보 대상인 객체를 참조하는 것을 관리(추가/제거) -> Subject 클래스로 일반화
    - addScore 메서드 : 각 통보 대상인 객체의 update메서드를 호출 -> observer 인터페이스로 일반화



1. ScoreRecord 클래스의 addScore(상태변경) 메소드 호출
   1. 자신의 성적 값
   2. 상태가 변경될 때마다 Subject 클래스의 notifyObservers메소드 호출
2. Subject 클래스의 notifyObservers 메서드 호출
   1. Observer 인터페이스를 통해 성적 변경을 통보
   2. DataSheetView 클래스의 update메서드 호출
   3. MinMaxView 클래스에 update 메서드 호출
3. Observer
   1. 추상화된 통보 대상
4. DataSheetView MinMaxView
   1. Observer를 Implements함으로써 구체적인 통보 대상이 됨
5. Subject
   1. 성적변경에 관심이 있는 대상 객체들을 관리
6. ScoreRecord
   1. Subject를 exteds함으로써 구체적인 통보 대상을 직접 참조하지 않아도 됨
7. 이렇게 Observer 패턴을 이용하면 ScoreRecord 클래스의 코드를 변경하지 않고도 새로운 관심 클래스 및 객체를 추가/제거하는 것이 가능해진다.

