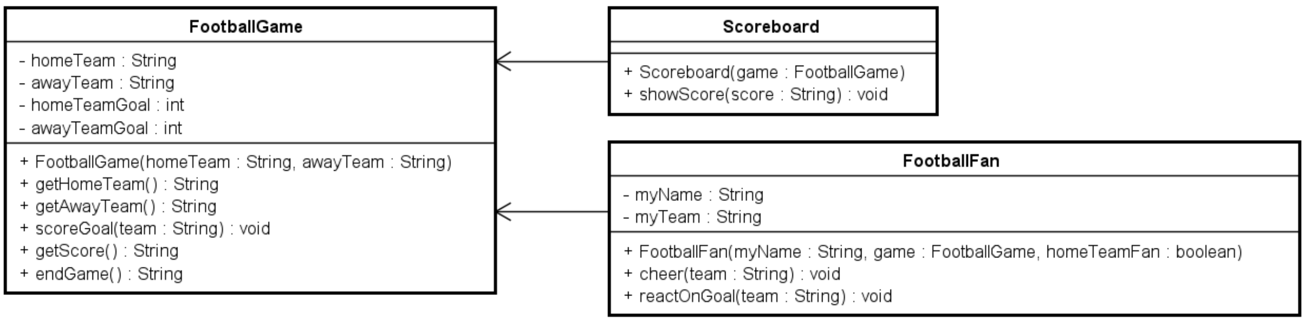
Exercises, Observer SDJ2

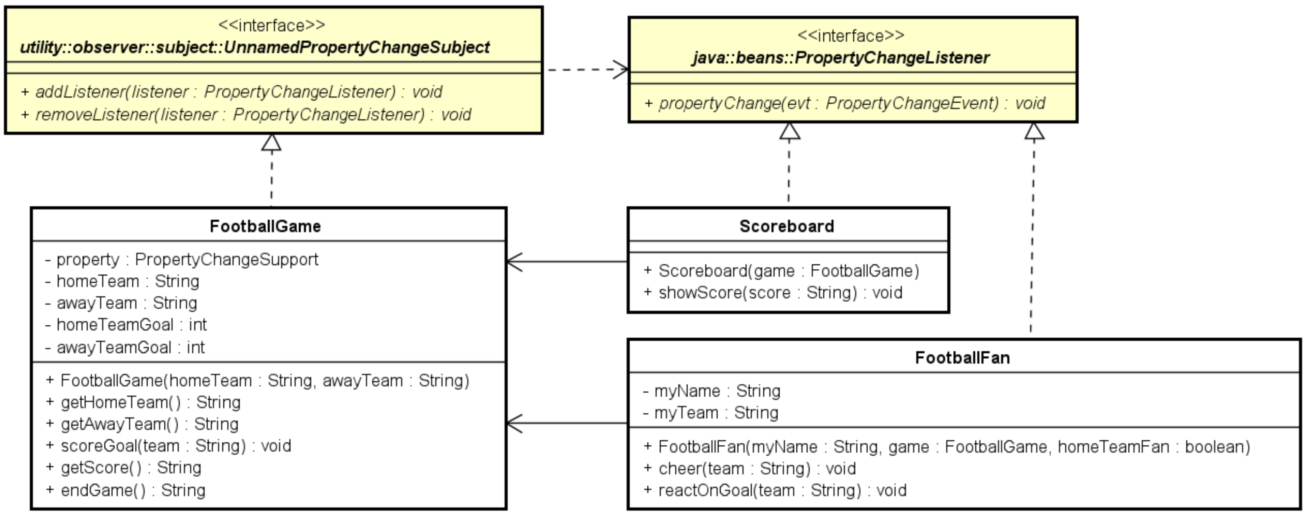
Exercise: Football game

A `FootBallGame` example is given below as a class diagram (source code can be found in appendix). If you run the main method given (in class `TestFootballGame`) you will see a simulation of a game. Note that Scoreboard and football fans only print out before the game starts and not during the game.


FootballGame
- homeTeam : String
- awayTeam : String
- homeTeamGoal : int
- awayTeamGoal : int
+ FootballGame(homeTeam : String, awayTeam : String)
+ getHomeTeam() : String
+ getAwayTeam() : String
+ scoreGoal(team : String) : void
+ getScore() : String
+ endGame() : String
Scoreboard
+ Scoreboard(game : FootballGame)
+ showScore(score : String) : void
FootballFan
- myName : String
- myTeam : String
+ FootballFan(myName : String, game : FootballGame, homeTeamFan : boolean)
+ cheer(team : String) : void
+ reactOnGoal(team : String) : void

Use this program as a template to implement the Observer Design Pattern as shown in the diagram below. Every time a team scores a goal, (method `ScoreGoal` in class `FootballGame` is called) the Scoreboard show the score and each football fan react depending on if their team scores or the other team scores.

Note: PropertyChangeListener is an interface from package java.beans in the API – do not declare this interface. However, interface UnnamedPropertyChangeSubject you either have to declare yourselves or import from package utility.observer.subject from the external jar file MyObserver-1.3.jar


<<interface>>
utility::observer::subject::UnnamedPropertyChangeSubject
+ addListener(listener : PropertyChangeListener) : void
+ removeListener(listener : PropertyChangeListener) : void
<<interface>>
java::beans::PropertyChangeListener
+ propertyChange(evt : PropertyChangeEvent) : void
FootballGame
- property : PropertyChangeSupport
- homeTeam : String
- awayTeam : String
- homeTeamGoal : int
- awayTeamGoal : int
+ FootballGame(homeTeam : String, awayTeam : String)
+ getHomeTeam() : String
+ getAwayTeam() : String
+ scoreGoal(team : String) : void
+ getScore() : String
+ endGame() : String
Scoreboard
+ Scoreboard(game : FootballGame)
+ showScore(score : String) : void
FootballFan
- myName : String
- myTeam : String
+ FootballFan(myName : String, game : FootballGame, homeTeamFan : boolean)
+ cheer(team : String) : void
+ reactOnGoal(team : String) : void

Extra:

Replace interface `UnnamedPropertyChangeSubject` with `NamedPropertyCangeSupport` as shown in the presentation (either from MyObserver1-3.jar or define it yourselves). Now you have to implement two methods taking a listener and a String (the property name they are listening to) – also simply by delegating to the `PropertyChangeSupport` instance variable.
Use the property name of the team scoring the goal when firing the event change and let the football fans only observe their own team (i.e. no message when the other team scores)

Appendix: Classes are given here:

Class FootballGame:

```
public class FootballGame
{
  private String homeTeam;
  private String awayTeam;
  private int homeTeamGoal;
  private int awayTeamGoal;

  public FootballGame(String homeTeam, String awayTeam)
  {
    this.homeTeam = homeTeam;
    this.awayTeam = awayTeam;
    this.homeTeamGoal = 0;
    this.awayTeamGoal = 0;
  }

  public String getHomeTeam()
  {
    return homeTeam;
  }

  public String getAwayTeam()
  {
    return awayTeam;
  }

  public void scoreGoal(String team)
  {
    if (team.equals(homeTeam))
    {
      homeTeamGoal++;
    }
    else if (team.equals(awayTeam))
    {
      awayTeamGoal++;
    }
  }

  public String getScore()
  {
    return homeTeamGoal + " - " + awayTeamGoal;
  }

  public String endGame()
  {
    return getScore();
  }

}
```

Class FootballFan:

```java
public class FootballFan
{
   private String myName;
   private String myTeam;
   private FootballGame game;

   public FootballFan(String myName, FootballGame game,boolean homeTeamFan)
   {
      this.myName = myName;
      this.game = game;
      if (homeTeamFan)
      {
         this.myTeam = game.getHomeTeam();
      }
      else
      {
         this.myTeam = game.getAwayTeam();
      }

      cheer(myTeam);
   }

   public void cheer(String team)
   {
      System.out.println(myName + "> Come on " + team);
   }

   public void reactOnGoal(String team)
   {
      if (team.equals(myTeam))
      {
         System.out.println(myName + "> Jubiiii (" + team + " scored)");
      }
      else
      {
         System.out.println(myName + "> Boooo (" + team + " scored)");
      }
   }

}
```

Class ScoreBoard:

```
public class Scoreboard
{
   private FootballGame game;

   public Scoreboard(FootballGame game)
   {
      this.game = game;
      showScore(game.getScore());
   }

   public void showScore(String score)
   {
      System.out.println("SCOREBOARD: " + score);
   }
}
```

Class GameSimulation:

```
import java.util.Scanner;

public class GameSimulation
{
  public static void main(String[] args)
  {
    FootballGame game = new FootballGame("Italy", "Portugal");

    Scoreboard scoreboard = new Scoreboard(game);

    addLocalFans(game);
    playTheGame(game);
  }

  private static void playTheGame(FootballGame game)
  {
    System.out.println(
        "Football game between " + game.getHomeTeam() + " and " + game
            .getAwayTeam() + ":");
    Scanner input = new Scanner(System.in);
    System.out.print("Press ENTER to start the game");
    input.nextLine();
    System.out.println("Game started");
    play();
    goalChance(game);
    play();
    goalChance(game);
    play();
    goalChance(game);
    play();
    goalChance(game);
    play();
    goalChance(game);

    String score = game.endGame();
    System.out.println("Game ended " + score);
  }

  private static void addLocalFans(FootballGame game)
  {
    FootballFan[] fans = new FootballFan[3];
    for (int i = 0; i < fans.length * 2 / 3; i++)
    {
      fans[i] = new FootballFan("Fan" + (i + 1), game, true);
    }
    for (int i = fans.length * 2 / 3; i < fans.length; i++)
    {
      fans[i] = new FootballFan("Fan" + (i + 1), game, false);
    }
  }

  private static void play()
  {
    try
    {
      Thread.sleep(5000);
    }
    catch (InterruptedException e)
    {
      // nothing
    }
  }

  private static void goalChance(FootballGame game)
  {
    String team = null;
    if ((int) (Math.random() * 2) == 0)
    {
      team = game.getHomeTeam();
    }
    else
    {
      team = game.getAwayTeam();
    }
    if ((int) (Math.random() * 4) > 0) // 75% chance
    {
      System.out.println("--->" + team + " scored a goal");
      game.scoreGoal(team);
    }
    else
    {
      System.out.println(team + " missed a shot");
    }
  }

}
```