

Die Aufgaben müssen einzeln bearbeitet und abgegeben werden. Die Abgabe muss bis **spätestens Donnerstag 22.11.2011 um 23:59 Uhr** über unser Hausaufgabenabgabesystem <http://seblog.cs.uni-kassel.de/pmws1213/> erfolgen. Die Abgabe ist nur als einzelne *.zip oder *.jar-Datei möglich. Daher müssen alle für eine Abgabe relevanten Daten zu einem solchen Archiv kombiniert werden. Diese Hausaufgabe gibt **17 + 6 Punkte**.

Hinweise zur Abgabe:

- Die Hausaufgabe als exportiertes Eclipse Projekt (*.zip, **nicht** den gesamten Workspace) abgeben. Das kann mit Hilfe der Eclipse Export Funktion durchgeführt werden. Ist das Projekt nicht korrekt exportiert, kann es bei der Korrektur nicht berücksichtigt werden (es bietet sich also an, den Import des exportierten Projektes auszuprobieren).

WICHTIG Benennen Sie ihre Projekte für diese und alle zukünftigen Abgaben nach folgendem Schema:

PMWS1213_HA<a>_A_<Matrikelnummer>,

wobei <a> für die aktuelle Hausaufgabe und für die Aufgabenummer steht. Beispiel:

PMWS1213_HA3_A1_12345678.

Aufgabe 1 - Implementierung des Klassendiagramms zu Diaballik (17P)

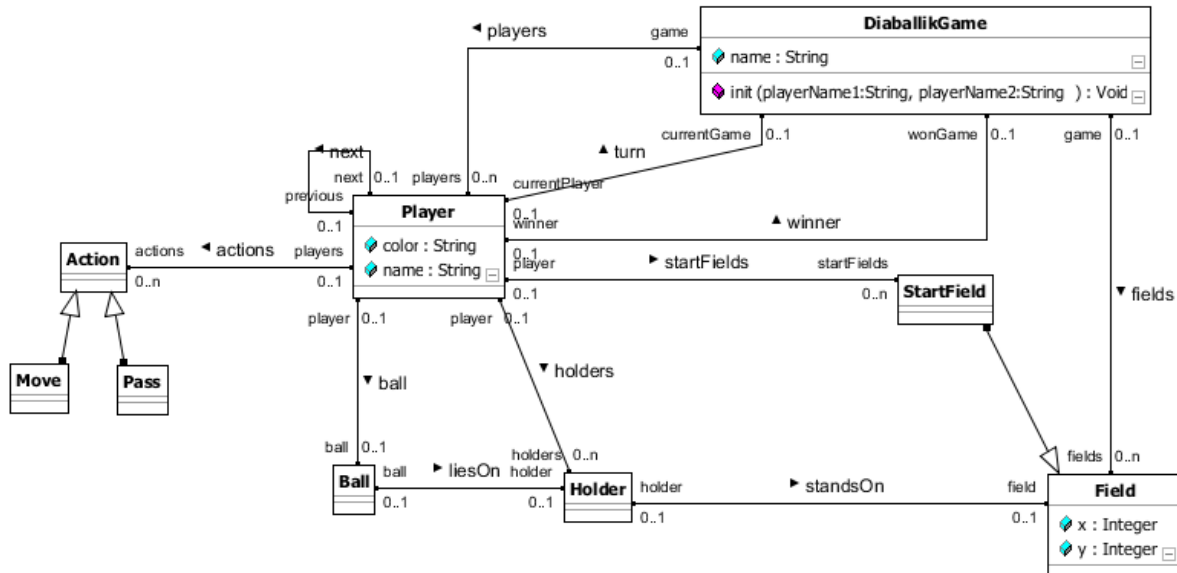


Abbildung 1: Diaballik Klassendiagramm

Gegeben ist das in Abbildung 1 dargestellte Klassendiagramm. Implementieren Sie das Klassendiagramm in Java. Folgende Vorgehensweise wird vorgeschlagen:

1. Erstellen Sie für jede Klasse im Diagramm eine 'public class' in einer separaten .java-Datei.
2. Fügen Sie den Klassen die entsprechenden Attribute hinzu. Achten Sie auf Verkapselung der Attribute durch getter/setter!
3. Fügen Sie den Klassen die entsprechenden Methoden hinzu - eine Implementierung der Methodenrumpfe ist nicht gefordert.
4. Implementieren Sie die Assoziationen und stellen Sie referenzielle Integrität sicher: Dazu fügen Sie den Klassen für zu-1-Assoziationen die Zugriffsmethoden set/get/with, für zu-n-Assoziationen die Zugriffsmethoden add/remove/get/with hinzu und wählen eine geeignete Containerklasse (z. B. `HashSet`). Sorgen Sie dafür, dass bei bidirektionalen Assoziationen die Rückrichtung automatisch mitgesetzt wird.

Zusatzaufgabe 2 - JUnit Tests (6P)

Schreiben Sie JUnit Tests, die die referenzielle Integrität der folgenden bidirektional Assoziationen testen:

- next (Player - Player)
- players (DiaballikGame - Player)
- standsOn (Holder - Field)

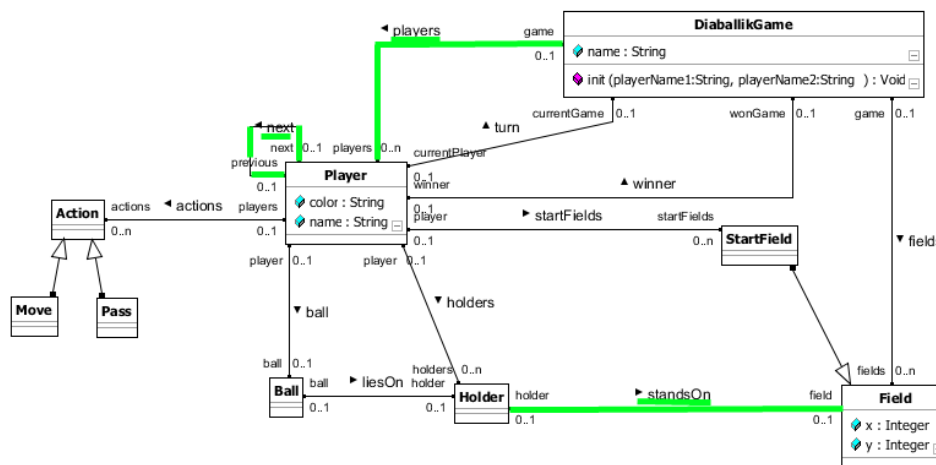


Abbildung 2: Diaballik Klassendiagramm mit hervorgehobenen Assoziationen

In Abbildung 2 sind diese drei Assoziationen noch einmal im Klassendiagramm farblich hervorgehoben.

Es soll sowohl das Setzen/Hinzufügen als auch das Löschen von Elementen überprüft werden. Schreiben Sie Tests die **beide** Richtungen der Assoziationen testen.

Hinweis: Weil die Tests zum Löschen eines Elements voraussetzen, dass das Element vorher hinzugefügt wurde, können diese in einem einzigen Test zusammengefasst werden. Sie benötigen also $3 \text{ (Assoziationen)} * 2 \text{ (Rollen)} = 6$ Tests. Nutzen Sie zur Verifizierung ihrer Tests passende assert-Methoden.

Bsp:

- `Assert.assertTrue(object.trueExpected());` bzw.
- `Assert.assertFalse(object.falseExpected());`
- `Assert.assertEquals(3, object.getSize());`