UNIKASSEL VERSITÄT

SOFTWARE ENGINEERING
UNIVERSITÄT KASSEL

Programmiermethodik

Übung 6

Wintersemester 2012 / 13

Fachgebiet Software Engineering

Tobias George

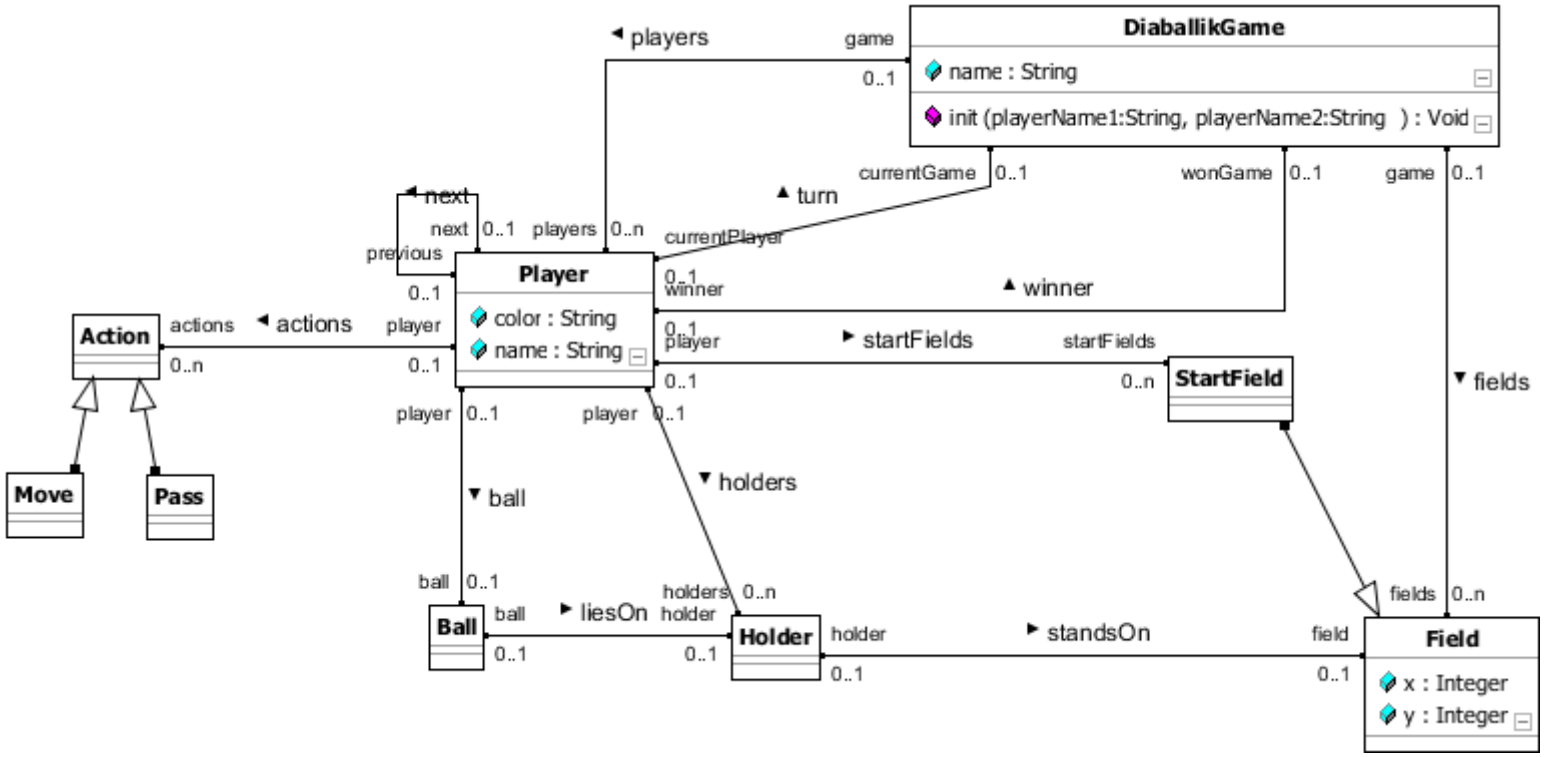
george@uni-kassel.de

Agenda

- Besprechung HA 4
- Fujaba4Eclipse: Storyboards
- SDMLib: Scenarios
- Praktische Übung: Storyboards/Scenarios
- Vorschau HA 5

Besprechung HA4 I

- **Aufgabe 1 (GERADE)– Klassendiagramm in Fujaba modellieren**



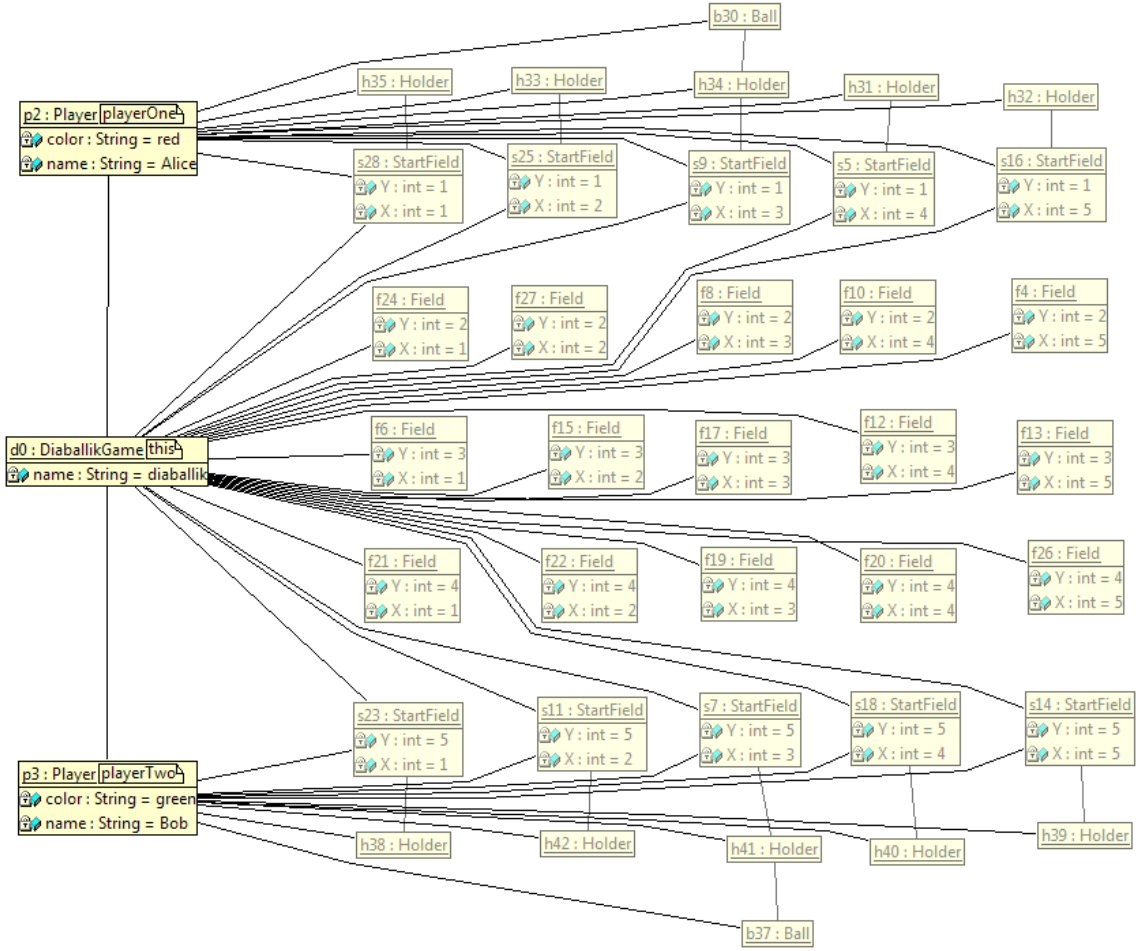
Besprechung HA4 II

- **Aufgabe 1 (UNGERADE) – Klassendiagramm in SDMLib modellieren**

```
private static final String INT = "int";
private static final String STRING = "String";

public static void main(String... args) {

    ClassModel model = new ClassModel("de.uniks.se");

    Clazz playerClazz = model.createClazz("Player",
            "color", STRING,
            "name", STRING);
    Clazz actionClazz = model.createClazz("Action");
    Clazz moveClazz = model.createClazz("Move");
    Clazz passClazz = model.createClazz("Pass");
    Clazz diaballikGameClazz = model.createClazz("DiaballikGame",
            "name", STRING);
    Clazz ballClazz = model.createClazz("Ball");
    Clazz holderClazz = model.createClazz("Holder");
    Clazz startFieldClazz = model.createClazz("StartField");
    Clazz fieldClazz = model.createClazz("Field",
            "X", INT, "Y", INT);

    diaballikGameClazz.withMethods("init(String,String)", "void");
    diaballikGameClazz.withMethods("checkEnd()", "void");
    holderClazz.withMethods("move(Field)","void");
    ballClazz.withMethods("pass(Holder)","void");

    moveClazz.withSuperClass(actionClazz);
    passClazz.withSuperClass(actionClazz);
    startFieldClazz.withSuperClass(fieldClazz);

    playerClazz.withAssoc(actionClazz, "actions", R.MANY, "player", R.ONE);
    playerClazz.withAssoc(playerClazz, "previous", R.ONE, "next", R.ONE);
    playerClazz.withAssoc(ballClazz, "ball", R.ONE, "player", R.ONE);
    playerClazz.withAssoc(holderClazz, "holders", R.MANY, "player", R.ONE);
    playerClazz.withAssoc(startFieldClazz, "startFields", R.MANY, "player", R.ONE);
    playerClazz.withAssoc(diaballikGameClazz, "wonGame", R.ONE, "winner", R.ONE);
    playerClazz.withAssoc(diaballikGameClazz, "currentGame", R.ONE, "currentPlayer", R.ONE);
    holderClazz.withAssoc(fieldClazz, "field", R.ONE, "holder", R.ONE);
    ballClazz.withAssoc(holderClazz, "holder", R.ONE, "ball", R.ONE);
    diaballikGameClazz.withAssoc(fieldClazz, "fields", R.MANY, "game", R.ONE);
    diaballikGameClazz.withAssoc(playerClazz, "players", R.MANY, "game", R.ONE);

    model.generate("genSrc");
    model.dumpClassDiag("genSrc", "diaballikClassDiag");
}
```

Besprechung HA5 V

- **Aufgabe 2 – eDOBS Screenshot nach `DiaballikGame:init(…)`**

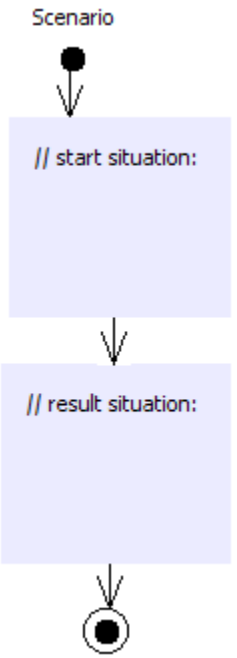


Fujaba Storyboards I

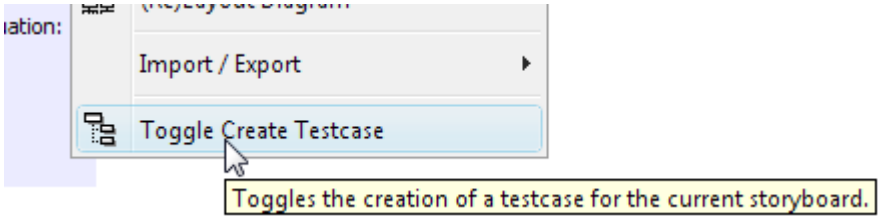
- **Tests schreiben ist aufwendig -> Storyboards**
- **„Closed World Assumption“: Objekte die nicht gezeichnet wurden gibt es nicht**
- **Ein Storyboard ist – wie ein „normaler“ JUnit Test auch – nicht allgemein gehalten sondern testet IMMER einen konkreten Fall**

Fujaba Storyboards II

- **Fujaba Model öffnen -> Diagrams -> New Story Board**
- **Start situation:**
  - Startsituation eines Szenarios herstellen
  - Methode aufrufen
- **Result situation:**
  - Endsituation prüfen

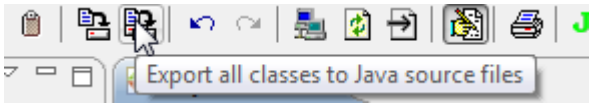


- **Wichtig: Rechtsklick auf das Diagramm -> Toggle Create Testcase**

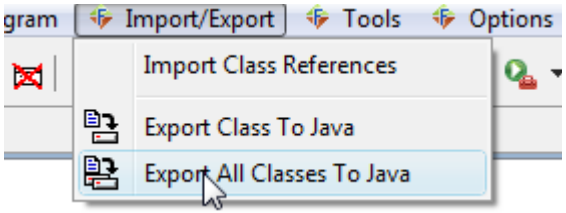


Fujaba Storyboards III

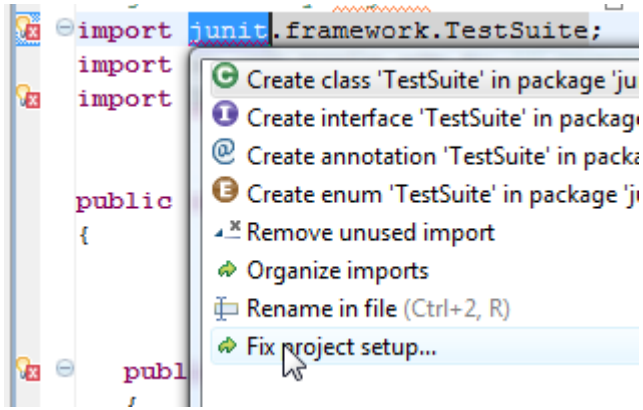
- **Wie gewohnt Code generieren**

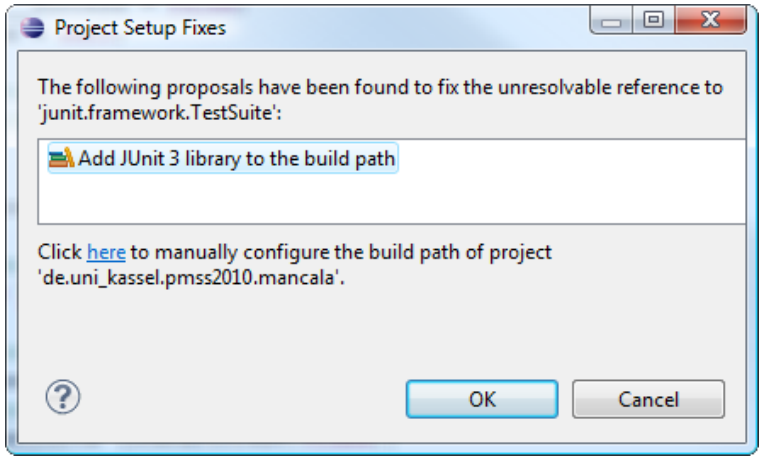


oder



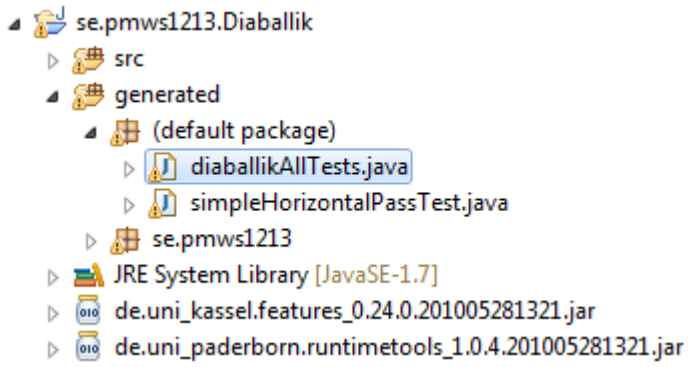
- **Neben dem Modell werden JUnit3 Testklassen generiert**
- **Falls nötig, JUnit3 bzw. JUnit4 Dependency hinzufügen**



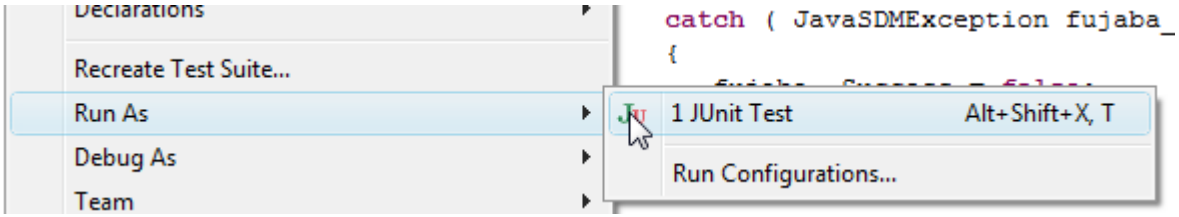


Fujaba Storyboards IV

- **Generierte Tests ausführen:**
  - Rechtsklick



  - Run As -> JUnit Test



SDMLib Scenarios I

- **Test manuell implementieren:**

```
Scenario scenario = new Scenario("src");

scenario.add("Start situation: ");

Room examRoom = new Room().withTopic("exsam").withCredits(0);
Room stochRoom = new Room().withTopic("stoch").withCredits(23);
Room calcRoom = new Room().withTopic("calc").withCredits(20)
      .withNeighbors(stochRoom);
Room mathRoom = new Room().withTopic("math").withCredits(17)
      .withNeighbors(calcRoom);
Room algRoom = new Room().withTopic("alg").withCredits(35)
      .withNeighbors(mathRoom);
Room philRoom = new Room().withTopic("phil").withCredits(32);
Room modRoom = new Room().withTopic("modeling").withCredits(29)
      .withNeighbors(examRoom)
      .withNeighbors(stochRoom)
      .withNeighbors(mathRoom)
      .withNeighbors(calcRoom);
Room artsRoom = new Room().withTopic("arts").withCredits(17)
      .withNeighbors(modRoom)
      .withNeighbors(philRoom);
...
```

- **Test Code mit Ausführen der Main-Methode generieren**
- **Test ausführen analog zu den von Fujaba generierten Tests**

Praktische Übung: Storyboards/Scenarios erstellen

- **Mancala Projekt vom PM Blog runterladen**
- **Storyboard/Scenario zu** Pit::moveStones() **erstellen**
  - Drei Pits,
  - Zwei Stones im ersten Pit
  - Aufruf der Methode Pit::moveStones()
- **Implementiert** Pit::moveStones() **manuell und testet die Methode mit dem generierten Test**

Vorschau HA6

- **Deadline: 06.12.2012, 23:59 Uhr**
- **Aufgabe 1: Storyboards mit Fujaba4Eclipse / Scenarios mit SDMLib:**
  - Roter Ball in grüner Endzone
  - Horizontales Bewegen eines Spielsteins
  - Diagonaler Pass
  - Vertikaler Pass über einen eigenen Spielstein

- **Randbedingungen:**
- 1 Spiel mit Namen ”diaballik”
- 2 Spieler, Alice (rot) und Bob (grün)
- In der Startsituation müssen zusätzlich lediglich die gelb umrandeten Informationen modelliert werden. Alle nicht benannten Objekte dürfen beliebige Objektnamen haben.

Ende

Jetzt: Betreutes Arbeiten

Ansonsten: Schönes WE!