The Splendid Charm of Badenshire

Abschlussveranstaltung

Manuel Geffken

Universität Freiburg, Germany

SS 2011

Inhalt

Willkommen zur Abschlussveranstaltung!

- Java-Projekt SS2011
- The Splendid Charm of Badenshire - ein 2D-Rollenspiel

Ablauf

- CRAHUL
  - Code Review
  - Ausschnitte: Highlights und Lowdarks
- Präsentation & Vorstellung Ihrer Projekte
  - Desktop (Gruppen 2, 4, 5, 6, 7)
  - Android (Gruppen 1, 3)
- Abstimmung & Preisverleihung

Das Java-Projekt 2011 in Zahlen (Stand 29.07.2011)

- 7 Teams (21 Teilnehmer)
  - 2 mal Android
  - 5 mal Desktop

Das Java-Projekt 2011 in Zahlen (Stand 29.07.2011)

- 7 Teams (21 Teilnehmer)
  - 2 mal Android
  - 5 mal Desktop
- 7 Teams (21 Teilnehmer) haben bestanden

Das Java-Projekt 2011 in Zahlen (Stand 29.07.2011)

- 7 Teams (21 Teilnehmer)
  - 2 mal Android
  - 5 mal Desktop
- 7 Teams (21 Teilnehmer) haben bestanden
- $>$696 Commits

Das Java-Projekt 2011 in Zahlen (Stand 29.07.2011)

- 7 Teams (21 Teilnehmer)
  - 2 mal Android
  - 5 mal Desktop
- 7 Teams (21 Teilnehmer) haben bestanden
- >696 Commits
- Source Lines of Code

Language	Files	Blank	Comment	Code
Java	186	2667	6308	10152
XML	12	3	0	312

Das Java-Projekt 2011 in Zahlen (Stand 29.07.2011)

- 7 Teams (21 Teilnehmer)
  - 2 mal Android
  - 5 mal Desktop
- 7 Teams (21 Teilnehmer) haben bestanden
- >696 Commits
- Source Lines of Code

Language	Files	Blank	Comment	Code
Java	186	2667	6308	10152
XML	12	3	0	312

- Praktische keine Unittests :-(

Top-Downn Review

- Paketstruktur

Top-Downn Review

- Paketstruktur
- Rekorde

Top-Downn Review

- Paketstruktur
- Rekorde
- Wichtige Klassen/Methoden/Datenstrukturen

Top-Downn Review

- Paketstruktur
- Rekorde
- Wichtige Klassen/Methoden/Datenstrukturen
- Kartenrepräsentationen

Paketstruktur

- Beispiel 1
  - game.Classes
  - game.Interfaces

Paketstruktur

- Beispiel 1
  - game.Classes
  - game.Interfaces
- Beispiel 2
  - de.proglang.javaNN.Background
  - de.proglang.javaNN.Basics
  - de.proglang.javaNN.Characters
  - de.proglang.javaNN.Characters

Paketstruktur

- Beispiel 1
  - game.Classes
  - game.Interfaces
- Beispiel 2
  - de.proglang.javaNN.Background
  - de.proglang.javaNN.Basics
  - de.proglang.javaNN.Characters
  - de.proglang.javaNN.Characters
- Beispiel 3
  - de.proglang.javaNN

Rekorde

- Die wenigsten Klassen: 7
  - Größte Klasse `game.Classes.GameWorld`
    - 921 LOC
    - 51 Felder

Rekorde

- Die wenigsten Klassen: 7
  - Größte Klasse `game.Classes.GameWorld`
    - 921 LOC
    - 51 Felder
- Die meisten Klassen: 46
  - Größte Klasse `de.proglang.javaNN.Maps.AKarte`
    - 198 LOC
    - 3 Felder

Rekorde

- Die wenigsten Klassen: 7
  - Größte Klasse `game.Classes.GameWorld`
    - **921 LOC**
    - 51 Felder
  - **LOC: 1683**
- Die meisten Klassen: 46
  - Größte Klasse `de.proglang.javaNN.Maps.AKarte`
    - **198 LOC**
    - 3 Felder
  - **LOC: 1753**

Rekorde

- Die wenigsten Klassen: 7
  - Größte Klasse `game.Classes.GameWorld`
    - **921 LOC**
    - 51 Felder
  - **LOC: 1683**
  - Größte Methode `GameWorld.onKeyPressed()`: 194 LOC
- Die meisten Klassen: 46
  - Größte Klasse `de.proglang.javaNN.Maps.AKarte`
    - **198 LOC**
    - 3 Felder
  - **LOC: 1753**
  - Größte Methode `AKarte.kachelnBuild()`: 62 LOC

onKeyPressed() in GameWorld - 194 LOC

```
package game.Classes;

/**
 * Class GameWorld, which manages everything
 */
public class GameWorld implements IWorld{
  @Override
  public void onKeyPressed(int keyCode, ISimulationController controller) {
    if(this.gameOver || this.won) {
      if(keyCode == 82) {
        restart();
      }
      //ESC-Key close the Game
      else if(keyCode == 27) {
        System.exit(1);
      }
    }
    else {
```

onKeyPressed() in GameWorld - fortgesetzt

```
//Difficulty switch
if(keyCode == 49) {
  this.ind = 1.0;
  setMonsters();
}
else if(keyCode == 50) {
  this.ind = 1.5;
  setMonsters();
}
else if(keyCode == 51) {
  this.ind = 2.0;
  setMonsters();
}
else if(keyCode == 52) {
  this.ind = 2.5;
  setMonsters();
}
// i-key for informations about key-bindings
else if(keyCode == 73) {
```

onKeyPressed() in GameWorld - fortgesetzt

```
        this.currMap.toggleInventory();
        if(this.currMap.getInventoryOpen()) {
          this.pause = true;
          this.infoOn = true;
        }
        else {
          this.pause = false;
          this.infoOn = false;
        }

      }
      // p-key toggle pause mode
      else if(keyCode == 80) {
        if(!this.currMap.getInventoryOpen())
          this.pause = !this.pause;
      }
      // s-key toggle weapon
      else if(keyCode == 83) {
        this.hero.toggleWeapon();
      }
```

onKeyPressed() in GameWorld - fortgesetzt

```
//Moving keys
// Arrow-up, move up
if(!this.pause) {
  if(!this.gameOver && !this.won) {
    if(keyCode == 17) {
      this.hold = !this.hold;
    }
  // Arrow-up, move up
  else if(keyCode == 38) {
    if(!this.hold) {
      this.hero.move(0, -1);
      this.heroLookDirection = up;
      this.hero.setDirection(up);
      // Tests if next field is a wall or an enemy
      if(isBarrier()) {
        this.hero.move(0, 1);
      }
    }
```

onKeyPressed() in GameWorld - fortgesetzt

```
        else {
          this.heroLookDirection = up;
          this.hero.setDirection(up);
        }
      }
      // Arrow-down, move down
      else if(keyCode == 40) {
        if(!this.hold) {
          this.hero.move(0, 1);
          this.hero.setDirection(down);
          this.heroLookDirection = down;
          // Tests if next field is a wall or an enemy
          if(isBarrier()) {
            this.hero.move(0, -1);
          }
        }
        else {
          this.hero.setDirection(down);
```

onKeyPressed() in GameWorld - fortgesetzt

```
            this.heroLookDirection = down;
        }
    }
    // Arrow-left, move left
    else if(keyCode == 37) {
        if(!this.hold) {
            this.hero.move(-1, 0);
            this.hero.setDirection(left);
            this.heroLookDirection = left;
            // Tests if next field is a wall or an enemy
            if(isBarrier()){
                this.hero.move(1, 0);
            }
        }
        else {
            this.hero.setDirection(left);
            this.heroLookDirection = left;
        }
```

onKeyPressed() in GameWorld - fortgesetzt

```
        }
        // Arrow−right, move right
        else if(keyCode == 39) {
          if(!this.hold) {
            this.hero.move(1, 0);
            this.hero.setDirection(right);
            this.heroLookDirection = right;
            // Tests if next field is a wall or an enemy
            if(isBarrier()){
              this.hero.move(−1, 0);
            }
          }
          else {
            this.hero.setDirection(right);
            this.heroLookDirection = right;
          }
        }
```

onKeyPressed() in GameWorld - fortgesetzt

```
// space−key, attack
else if(keyCode == 32) {
  hit();
  this.hit = true;
}
// Test if current field is a door
int curX = this.hero.getXPos();
int curY = this.hero.getYPos();

// forwards
if(this.currMap.isDoorForward(curX, curY)) {
  if(this.hero.gotKey(currMap, 0)) {
    if(currMapNo != 4) {
      this.currMapNo++;
      if(currMapNo == 1)
        this.currMap = this.mapOne;
      if(currMapNo == 2)
        this.currMap = this.mapTwo;
```

onKeyPressed() in GameWorld - fortgesetzt

```
        if(currMapNo == 3)
          this.currMap = this.mapThree;
        if(currMapNo == 4)
          this.currMap = this.mapFour;
        if (this.currMapNo == 1)
        {
          this.hero.setXPos(1);
          this.hero.setYPos(1);
          this.hero.setDirection(down);
          this.heroLookDirection = down;
        }
        else
        {
          this.hero.setXPos(1);
          this.hero.setYPos(7);
        }

      }
```

onKeyPressed() in GameWorld - fortgesetzt

```
            else {
              if(this.heroLookDirection == right) {
                this.hero.setXPos(16);
                this.hero.setYPos(13);
              }
              else {
                this.hero.setXPos(14);
                this.hero.setYPos(13);
              }
            }
          }

          else
            this.hero.move(-1, 0);
        }
```

onKeyPressed() in GameWorld - fortgesetzt

```
// backwards
if(this.currMap.isDoorBackward(curX, curY)) {
  if(this.hero.gotKey(currMap, 1)) {
    this.currMapNo−−;
    if(currMapNo == 1)
      this.currMap = this.mapOne;
    if(currMapNo == 2)
      this.currMap = this.mapTwo;
    if(currMapNo == 3)
      this.currMap = this.mapThree;
    if(currMapNo == 4)
      this.currMap = this.mapFour;
    this.hero.setXPos(18);
    this.hero.setYPos(7);
  }
  else
    this.hero.move(1, 0);
  }
}
```

onKeyPressed() in GameWorld - fortgesetzt

```
    }
  }
  // Test if a item is on current position
  if(currMap.isItem(this.hero.getXPos(), this.hero.getYPos())){
    int curX = this.hero.getXPos();
    int curY = this.hero.getYPos();
    Item item = currMap.getItem(curX, curY);
    if(item.getItemNo() == 0) {
      this.hero.putHealth(item.heal());
      this.currMap.deleteItem(curX, curY);
    }
    else if(item.getItemNo() == 1 || item.getItemNo() == 2) {
      this.hero.putWeapon(this.currMap.getItem(curX, curY));
      this.currMap.deleteItem(curX, curY);
    }
```

onKeyPressed() in GameWorld - fortgesetzt

```
        else if(item.getItemNo() == 3) {
          this.hero.putKey(this.currMap.getItem(curX, curY));
          this.currMap.deleteItem(curX, curY);
        }
        else if(item.getItemNo() == 4) {
          this.hero.putItem(item);
          this.currMap.deleteItem(curX, curY);
        }
        else if(item.getItemNo() == 5) {
          this.currMap.deleteItem(curX, curY);
          this.won = true;
        }

      }
    }
}
```

Felder in GameWorld (Auszug)

```
/**
 * Map Zero
 */
private Map mapZero;
/**
 * Map one
 */
private Map mapOne;
/**
 * Map two
 */
private Map mapTwo;
/**
 * Map three
 */
private Map mapThree;
/**
 * Map four
 */
private Map mapFour;
```

Felder in GameWorld (Auszug)

```
/**
 * Monster of map zero
 */
private Monster[] monsterMapZero = new Monster[1];
/**
 * Monsters of map one
 */
private Monster[] monsterMapOne = new Monster[15];
/**
 * Monsters of map two
 */
private Monster[] monsterMapTwo = new Monster[15];
/**
 * Monsters of map three
 */
private Monster[] monsterMapThree = new Monster[15];
/**
 * Monsters of map four
 */
private Monster[] monsterMapFour = new Monster[30];
```

Felder in GameWorld (Auszug)

```
/**
 * Key three
 */
private Item keyThree = new Item(3, 2, 5, "keyThree");
/**
 * Key four
 */
private Item keyFour = new Item(4, 12, 8, "keyFour");
/**
 * Helmet
 */
private Item helmet = new Item("1", 30, 1, 1 , 3);
/**
 * Shield
 */
private Item shield = new Item("", 30, 16, 13, 1);
```

AKarte.kachelnBuild() - 62 LOC

```
public abstract class AKarte implements IKarte {
  protected void kachelnBuild (MapEvents ereignisse, String kachelString, IInit init){
    int charcount = 0;
    for(int k = 0; k < Konstanten.ANZKACHELNY; ++k){
      for(int i = 0; i < Konstanten.ANZKACHELNX; ++i){
        switch(kachelString.charAt(charcount)){
          case '#':
            this.alleKacheln [i][k] = new Steinwand(ereignisse, i, k, init);
            break;
            ...
          default:
            this.alleKacheln [i][k] = new EinfacherBoden(ereignisse, i, k, init);
            break;
        }
        ++charcount;
      }
    }
  }
}
```

Alle Felder in `AKarte`

```
public abstract class AKarte implements IKarte {
  /**
   * Alle Kacheln einer Karte Erster Index: x-Koordinate; Zweiter Index
   * y-Koordinate
   */
  protected IKachel[][] alleKacheln;

  /**
   * Das IInit-Objekt
   */
  protected IInit init;

  /**
   * Die Monster auf dieser Karte
   */
  protected Collection<ICharacter> alleMonster;
}
```

Rekorde

- Die wenigsten Codezeilen: 1324
  - Größte Klasse `de.proglang.Player`: 258 Zeilen

Rekorde

- Die wenigsten Codezeilen: 1324
  - Größte Klasse `de.proglang.Player`: 258 Zeilen
- Die meisten Codezeilen: 1753
  - Größte Klasse `de.proglang.javaXX.Maps.AKarte`: 198 Zeilen

Rekorde

- Die wenigsten Codezeilen: 1324
  - Größte Klasse `de.proglang.Player`: 258 Zeilen
- Die meisten Codezeilen: 1753
  - Größte Klasse `de.proglang.javaXX.Maps.AKarte`: 198 Zeilen
- Die wenigsten Commits: 14

Rekorde

- Die wenigsten Codezeilen: 1324
  - Größte Klasse `de.proglang.Player`: 258 Zeilen
- Die meisten Codezeilen: 1753
  - Größte Klasse `de.proglang.javaXX.Maps.AKarte`: 198 Zeilen
- Die wenigsten Commits: 14
- Die meisten Commits: 223

Kartenrepräsentationen

```
77777777777777777777777777777777777777
722271d22711714444444441111111111111117
722271722711745555555551777777777777717
722271722711745555555551722222222222717
7222d1722711745555555551722222222222717
777771777711745551115551722222222797717
71111111d111115551N1555172222222272717
71111111711111555111555172222727222222717
71111111777771555555555172222M222222717
71111111722271555555555172222727222222717
71111111722271555555555172222222222222717
71111111d22271111111111172222222222e17
77777777777777777O7777778777777777b7b77
71111b1111111111171171N11711111a111111117
71111111111111171171111171111111311317
...
```

Kartenrepräsentationen

```
# # # # # # # # # # # # # # # #
# . . . . . . . . . . . . . . #
# . . . . # . # # # . # # # # #
# . . # . # . . . # . # . . . +
# . # # . # . # # # . # # # # #
# . # . . # . # . # . . . . . #
# . # . . . . # . # # # # # . #
# . # . # # # # . . . . . # . #
# . . . . . . # . # . # . # . #
# . # # # # . # . # . # . # . +
# . . . . . . . . # . . . . . #
# # # # # # # # # # # # # # # #
```

Kartenrepräsentationen

```
"############"
"#g-#---#-#s#"
"#--+-#-+-#-#"
"#--#---+-+-#"
"#--###-#++-#"
"#----+-+-+-#"
"#+#+#+#--###"
"#-----#--+-#"
"###+#+#-#--#"
"##----+-#--#"
"#--#-#+-+--#"
"##-----##+-#"
"#----#+--+-#"
"##+#--+--###"
"#---+#-#--#"
"############"
```

Kartenrepräsentationen

```
xxxxxxxxxxxxxxxxxxxx
x---x---x---x------x
x-x-x-x---x-x-xx---x
x-x---x---x---x----x
x-xxxxxxxxxxxx----x
x----------x-------x
xxxxxxxxx--x-------x
x-------x--x-------x
x----------x-------x
x----------x-------x
x----------x-------x
x----------x-------x
xxxxxxxxxxxxxxxxxxxx
```

Kartenrepräsentationen

```
###level 1###
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
1,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,16:l,6,1
1,1,0,0,0,0,1,1,1,1,1,1,0,0,0,1,1,1,1,1
1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0,0,0,1,1
1,1,1,1,0,1,0,0,1,1,1,1,0,0,1,1,1,0,1,1
1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1
1,0,0,0,0,4:3,16:r,0,0,0,0,0,0,0,1,1,1,1,1,1
1,0,0,0,0,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1
1,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,1
1,0,7,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,3:1
1,1,1,0,0,1,1,1,0,0,0,0,1,1,1,1,1,1,1,1,1
1,0,0,0,0,1,1,1,0,0,0,0,1,1,1,1,1,1,1,1,1
1,12:1,0,0,0,1,1,1,0,0,0,0,1,0,0,0,0,0,0,1
1,1,1,1,1,1,1,1,1,1,1,1,0,0,11:2,0,0,17:r,0,0,8,1
1,1,1,1,1,1,1,1,1,1,1,1,3:2,1,1,1,1,1,1,1,1,1
###end###
```

Kartenrepräsentationen

```
1,1,1,1,1,1,1,2,1,1,1,1,1,1,1,1
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1
1,2,2,1,2,2,2,2,2,2,2,2,1,2,2,1
1,2,2,1,2,2,2,2,2,2,2,2,1,2,2,1
1,2,2,1,2,2,2,2,2,2,2,2,1,2,2,1
1,2,2,1,2,2,2,2,2,2,2,2,1,2,2,1
1,2,2,1,2,2,2,2,2,2,2,2,1,2,2,1
1,2,2,1,2,2,2,2,2,2,2,2,1,2,2,1
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
```

Kartenrepräsentationen

```
public void drawMapThree() {
  for (int i = 0; i < 20; i++) {
    for (int j = 0; j < 15; j++) {
      if(i == 2 && j > 0 && j < 13 && j != 5)
        this.fields[i][j] = 5;
      if(j == 6 && i > 0 && i < 9)
        this.fields[i][j] = 5;
      if(i == 4 && j > 7 && j < 14)
        this.fields[i][j] = 5;
      if(i == 9 && j > 1 && j < 13)
        this.fields[i][j] = 5;
      if(j == 8 && i > 9 && i < 20 && i != 16)
        this.fields[i][j] = 5;
      if(j == 2 && i > 9 && i < 19 && i != 18)
        this.fields[i][j] = 5;
      ...
    }
  }
}
```

Präsentation der Projekte

- Desktop
  - Gruppe 2
  - Gruppe 4
  - Gruppe 5
  - Gruppe 6
  - Gruppe 7
- Android
  - Gruppe 1
  - Gruppe 3

Abstimmung & Preisverleihung

- Welches Spiel sieht am besten aus?
- Höchster Spaßfaktor?