Dietrich Boles
Universität Oldenburg

Droidfoot
–
Portierung von Greenfoot-Szenarien nach Android

Version 2.0

1 Einleitung

Greenfoot (www.greenfoot.org) ist eine interaktive Java-Entwicklungsumgebung, die primär für Ausbildungszwecke entwickelt wurde. Sie erlaubt die einfache Entwicklung zweidimensionaler graphischer Anwendungen wie z. B. Simulationen und Spiele. Mit Greenfoot entwickelte Programme (Szenarien genannt) können als JavaSE-Anwendungen auf dem PC als auch als Applet in Web-Browser ausgeführt werden.

Droidfoot (http://www.programmierkurs-java.de/droidfoot) ist als Projekt zu verstehen, das es ermöglicht, erstellte Greenfoot-Szenarien auch als Android-Apps auf Android-Smartphones ausführen zu können. Droidfoot besteht dabei aus der Greenfoot-Klassenbibliothek, die nach Android portiert wurde, sowie einem Player, mit dem die Szenarien auf dem Smartphone ausgeführt werden können. Unterstützt wird dabei Android ab der Version 4.0.

1.1 Änderungen von Version 2.0 gegenüber Version 1.0

Gegenüber der Version 1.0 weist die Version 2.0 von Droidfoot folgende Änderungen auf:

- Die Greenfoot-Klasse *GreenfootSound* wurde implementiert (siehe Kapitel 2.6).
- Die Greenfoot-Klasse *UserInfo* wurde implementiert (siehe Kapitel 2.7).
- Es wird nun auch die Rotation von Akteuren bei der Implementierung der Touch- und Kollisionserkennungsmethoden berücksichtigt.

- Es ist nun auf einfache Art und Weise möglich, bestimmte Anpassungen am Android-Greenfoot-Player vorzunehmen (siehe Kapitel 5.4).
- Die Beispielszenarien wurden leicht modifiziert (siehe Kapitel 4).

1.2 Aufbau des Handbuches

In diesem Dokument werden zunächst in Kapitel 2 die Änderungen der Android-Greenfoot-Klassenbibliothek gegenüber der Original-Greenfoot-Klassenbibliothek beschrieben. Kapitel 3 führt den grundsätzlichen Aufbau und die Funktionsweise des Android-Greenfoot-Players ein. Kapitel 4 widmet sich der Installation von Droidfoot, wenn Sie Ihre eigenen Greenfoot-Szenarien als Android-Apps umsetzen möchten. Kapitel 5 schließlich stellt die grundsätzliche Vorgehensweise vor, um bereits fertige Greenfoot-Szenarien als Android-Apps zu realisieren.

2 Greenfoot-Klassenbibliothek für Android

Die Greenfoot-Klassenbibliothek besteht aus den Klassen:

- *Actor*
- *Greenfoot*
- *GreenfootImage*
- *GreenfootSound*
- *MouseInfo*
- *UserInfo*
- *World*

Ab der Version 2.0 von Droidfoot werden alle Klassen unterstützt.

Für das JavaSE-Paket *java.awt*, das in Android nicht verfügbar ist, wurde mit dem Paket *greenfoot.awt* ein Ersatzpaket definiert. Es stellt die Klassen *Color* und *Font* als Ersatz für die entsprechenden AWT-Klassen zur Verfügung.

2.1 Greenfoot

Folgende Methoden der Klasse *Greenfoot* werden in Droidfoot nicht unterstützt:

- *getMicLevel*
- *getKey*
- *isKeyDown*
- *mouseMoved*
- *mouseClicked*

Die Methode *mousePressed* liefert *true* im Falle eines Touch-Events anstelle des entsprechenden Mouse-Pressed-Events.

2.2 World

Die Methoden, die in ihrer Signatur die Klasse *java.awt.Color* benutzen, wurden an die Klasse *greenfoot.awt.Color* angepasst.

Bei der Definition einer eigenen World-Klasse ist zu beachten,

- dass diese immer einen parameterlosen Konstruktor besitzen muss und
- dass in jedem Konstruktor eine der *setBackground*-Methoden aufgerufen werden muss.

2.3 Actor

Bei der Definition einer Actor-Klasse ist zu beachten,

- dass in jedem Konstruktor eine der *setImage*-Methoden aufgerufen werden muss.

2.4 GreenfootImage

Folgende Methoden der Klasse *GreenfootImage* werden in Droidfoot nicht unterstützt:

- *drawShape*
- *fillShape*

Die Methoden, die in ihrer Signatur die Klasse *java.awt.Color* oder die Klasse *java.awt.Font* benutzen, wurden an die Klassen *greenfoot.awt.Color* bzw. *greenfoot.awt.Font* angepasst.

Die Methode *getAwtImage* besitzt als Funktionstyp die Klasse *android.graphics.Bitmap.*

2.5 MouseInfo

Die Schnittstelle der Klasse *MouseInfo* ist unverändert. Allerdings werden in Droidfoot keine Doppelklicks unterstützt, d.h. die Methode *getClickCount* liefert immer den Wert 1.

2.6 GreenfootSound

Alle Methoden der Klasse *GreenfootSound* werden in Droidfoot unterstützt. Die Umsetzung ist mit Hilfe des Android-MediaPlayers erfolgt, womit alle Audio-Formate unterstützt werden, die dieser unterstützt (siehe http://developer.android.com/guide/appendix/media-formats.html).

2.7 UserInfo

Alle Methoden der Klasse *UserInfo* werden in Droidfoot unterstützt. Die Speicherung und Verwaltung der Daten erfolgt nicht auf einem Server sondern lediglich lokal auf dem Android-Gerät, also analog zu dem Fall, dass ein Szenario in Greenfoot selbst ausgeführt wird. Der Benutzername kann im Android-Greenfoot-Player unter „Einstellungen“ geändert werden. Die Methode *isStorageAvailable* liefert genau dann *false*, wenn der Benutzername leer ist.

2.8 Klasse greenfoot.awt.Color

Die Klasse *greenfoot.awt.Color* dient als Ersatz der Klasse *java.awt.Color*. Die neu defnierte Klasse bietet dabei weitgehend dieselben Funktionalitäten.

In Android werden Farbinformationen grundsätzlich in einem int-Wert kodiert (siehe Klasse *android.graphics.Color*). Dieser int-Wert kann mit Hilfe der Methode *getColor* bzw. *getRGB* ermittelt werden.

2.9 Klasse greenfoot.awt.Font

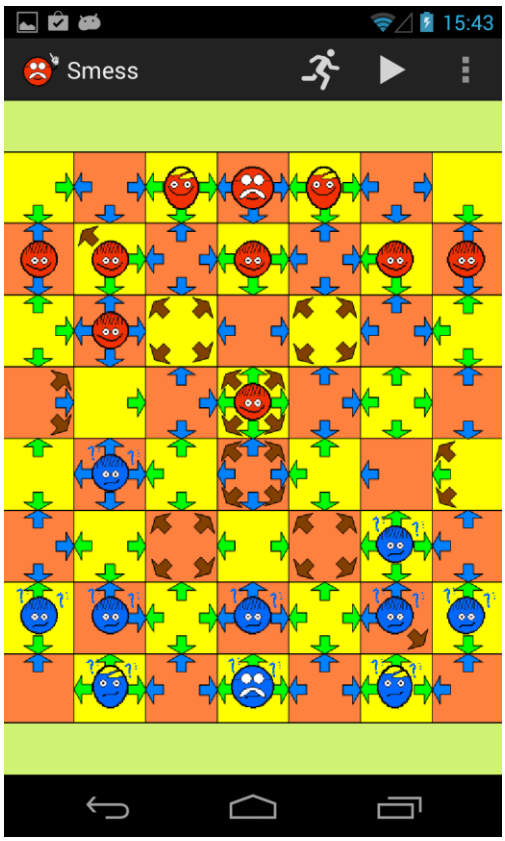
Die Klasse *greenfoot.awt.Font* dient als Ersatz der Klasse *java.awt.Font*. In Android werden Fonts anders gehandhabt als in JavaSE-AWT. Intern werden in der neuen Font-Klasse Fonts durch ein Objekt der Klasse *android.graphics.Typespace* und einen int-Wert für die Font-Größe repräsentiert.

Von daher ist diese Klasse mit Vorsicht einzusetzen. Sollten Sie extensiven Gebrauch von Fonts in ihren Greenfoot-Szenarien machen, ist eine Umstellung auf die Klasse *android.graphics.Typespace* anzuraten und der entsprechende Konstruktor der neuen Klasse *Font* zu nutzen.

3 Greenfoot-Player für Android

Voraussetzung zur Nutzung des Android-Greenfoot-Players ist ein Android-Smartphone mit einer Android-Version ab 4.0. Grund hierfür ist die Verwendung der ActionBar für die Unterbringung aller Steuerelemente (siehe auch http://developer.android.com/guide/topics/ui/actionbar.html).

Die folgende Abbildung skizziert die Gestaltung des Greenfoot-Players für Android.



Der komplette Bildschirmbereich ist für die Darstellung der Greenfoot-Welt reserviert. Oben befindet sich die ActionBar mit den Steuerelementen. Links in der ActionBar befinden sich das App-Icon sowie der App-Name. Rechts in der ActionBar bzw. bei Platzmangel im Overflow-Bereich sind die Aktions-Buttons bzw. Aktions-Items angeordnet. Es existieren - wenn nicht entsprechend angepasst (siehe Kapitel 5.4) - folgende Aktionen:

- Geschwindigkeit (Läufer-Symbol): Beim Anklicken erscheint in der Toolbar ein Slider, mit dem die Geschwindigkeit eingestellt werden kann. Beim Touch auf das App-Icon verschwindet der Slider wieder.
- Start/Pause (Start/Pause-Symbol): Beim Anklicken des Start/Pause-Items startet bzw. pausiert das Greenfoot-Szenario (äquivalent zum Greenfoot-Start/Pause-Button).

- Neues Spiel (Reset-Symbol): Beim Anklicken des „Neues Spiel“-Items wird die Greenfoot-World neu erzeugt (äquivalent zum Greenfoot-Reset-Button).
- Hilfe: Beim Anklicken des Hilfe-Items öffnet sich ein Web-Browser mit einer Hilfe-Seite.
- Einstellungen: Beim Anklicken des Einstellungen-Items erscheint eine Seite mit folgenden Einstellungsmöglichkeiten:
    - Skalierung: Beim Aktivieren der Checkbox wird die Welt immer auf Bildschirmgröße skaliert. Beim Deaktivieren ist es möglich, dass die Welt nur einen Teil des Bildschirms füllt.
    - Bildschirmausrichtung: Beim Klick auf dieses Item erscheint eine Dialogbox, in der festgelegt werden kann, ob die Darstellung immer im Landscape- oder Portrait-Modus erfolgen soll oder ein automatischer Wechsel (je nach Lage des Smartphones) erlaubt wird.
    - Name: Beim Klick auf dieses Item erscheint eine Dialogbox, in der der Benutzername geändert werden kann.

  Durch einen Benutzer durchgeführte Änderungen der Einstellungen bleiben auch über das Ende der Ausführung der App hinaus bestehen.

4 Installation

Voraussetzung zur Nutzung von Droidfoot ist die Installation von *Eclipse* und dem *Android SDK*. Lesen Sie sich dazu bitte die entsprechenden Informationen auf der Android-Developer-Website (http://developer.android.com/sdk/index.html) durch und folgen Sie den dortigen Ausführungen. Es ist sicher von Vorteil, wenn Sie bereits einmal eine, wenn auch kleine, eigene Android-App entwickelt haben.

Laden Sie von der Droidfoot-Website die Datei *droidfoot-2.0.zip* herunter und entpacken Sie diese. Der enthaltene Ordner *eclipse-workspace-droidfoot* kann direkt als Eclipse-Workspace genutzt werden. Er enthält folgende Projekte:

- *Droidfoot*: Ein Android Library Projekt, dass die Android-Greenfoot-Klassenbibliothek und den Android-Greenfoot-Player enthält (inkl. Quellcode und Ressourcen). Dieses Projekt muss als Android Library in alle neuen Android-Greenfoot-Projekte eingebunden werden.
- *DFTemplate*: Ein Android Application Projekt, das als Vorlage bei der Entwicklung neuer Android-Greenfoot-Szenarien dient.
- *DFBalloons*: Ein Beispiel-Projekt, das die Android-Umsetzung des Standard-Greenfoot-Szenarios Balloons enthält.
- *DFSnake*: Ein Beispiel-Projekt, das eine Android-Umsetzung des bekannten Snake-Spiels enthält (siehe auch http://www.greenfoot.org/scenarios/9260).
- *DFLights*: Ein Beispiel-Projekt, das die Android-Umsetzung des bekannten Lights-Knobelspiels enthält (siehe auch http://www.greenfoot.org/scenarios/1380).
- *DFSmess*: Ein Beispiel-Projekt, das die Android-Umsetzung eines schach-ähnliches Spiels namens *Smess* enthält (siehe auch http://www.greenfoot.org/scenarios/354).

Starten Sie Eclipse und wechseln Sie zum Workspace *eclipse-workspace-droidfoot* (> File > Switch Workspace)

5 Vorgehensweise

In diesem Kapitel wird die generelle Vorgehensweise bei der Portierung eines Greenfoot-Szenarios nach Android beschrieben. Beachten Sie bitte im Vorfeld die Unterschiede zwischen der Original-Greenfoot-Bibliothek und der Android-Greenfoot-Bibliothek (siehe Kapitel 2). Beispielsweise unterstützt die Android-Greenfoot-Bibliothek keine Tastatureingaben.

5.1 Kürzester Weg

Als Beispiel für die Skizzierung der Portierungsvorgehensweise wird im Folgenden die Portierung eines Greenfoot-Szenarios namens *Snake* genutzt, dass das allgemein bekannte Snake-Spiel umsetzt.

1. Copy-Pasten Sie das Eclipse-Android-Projekt *DFTemplate* in ein neues Eclipse-Projekt namens *Snake* (für dieses Beispiel).
2. Achten Sie darauf, dass das Eclipse-Android-Projekt Droidfoot als Library eingebunden ist (> Properties > Android), was nach Ausführung von Schritt 1 standardmäßig der Fall ist.
3. Öffnen Sie die Dateien *strings.xml* in dem Ordner *res/values* und ersetzen Sie den dort eingetragenen Namen *DFTemplate* durch den Namen Ihres Szenarios (bspw. *Snake*).
4. Nennen Sie im Ordner *src* das Paket *org.droidfoot.dftemplate* um (bspw. *org.droidfoot.snake*) (> Refactor > Rename).
5. Nennen Sie im Ordner *src* im in Schritt 4 umbenannten Paket die Datei/Klasse *DTTemplateActivity* um (bspw. *SnakeActivity*)
6. Benennen Sie in der Datei *AndroidManifest.xml* das package-Attribut entsprechend Schritt 3 um (2. Zeile) (bspw. *package="org.droidfoot.snake"*)
7. Kopieren Sie die java-Dateien Ihres Greenfoot-Szenarios in das in Schritt 4 umbenannte Paket. Beachten Sie, dass alle java-Dateien als erste Anweisung die entsprechende package-Anweisung benötigen (*package org.droidfoot.snake;*).
8. Beseitigen Sie Fehler, die aufgrund der Unterschiede der Original-Greenfoot-Klassenbibliothek und der Android-Greenfoot-Klassenbibliothek bestehen (siehe Kapitel 2).
9. Öffnen Sie die in Schritt 5 umbenannte Activity-Datei, entfernen Sie dort den Kommentar der auskommentierten Anweisung in der Methode *onCreate* und weisen Sie der Variablen *DroidfootActivity.worldClass* das Klassenobjekt der World-Klasse Ihres Greenfoot-Szenarios zu (*DroidfootActivity.worldClass = SnakeWorld.class;*)
10. Kopieren Sie die Ordner *images* und *sounds* Ihres Greenfoot-Szenarios, die die Bilder und Sounds enthalten, in den Ordner *assets*.
11. Starten Sie das Projekt als Android Application (> Run as > Android Application)!

5.2 Eigene Launcher-Icons

Standardmäßig werden die Launcher-Icons der Android-Library *Droidfoot* genutzt. Sie können jedoch auch für Ihre Greenfoot-App ein eigenes Icon verwenden. Kopieren Sie dazu, wie unter Android üblich, die entsprechenden Icon-Dateien (png oder gif) in die entsprechenden *drawable*-Unterordner des Ordners *res*. Beachten Sie, dass die Icon-Dateien den Namen *droidfoot_launcher* (mit Endung .png oder .gif) besitzen müssen!

5.3 Eigene Hilfe-Dateien

Standardmäßig wird beim Aktivieren des Hilfe-Items im ActionBar-Menü eine Seite mit „keine Hilfe verfügbar“ angezeigt. Sie können jedoch eine Hilfeseite für Ihr Android-Greenfoot-Szenario anlegen.

Erstellen Sie dazu eine html-Datei mit dem Namen *help.html* und platzieren Sie diese in den Ordner *res/raw*. Für eine mehrsprachige lokalisierte Hilfe beachten Sie die entsprechenden Android-Regeln. Platzieren Sie bspw. eine englische Hilfe-Datei in den Ordner *res/raw* und eine deutsche Hilfe-Seite in den Ordner *res/raw-de*.

5.4 Anpassungen des Android-Greenfoot-Players

Seit der Version 2.0 von Droidfoot ist es in der eigenen Activity-Datei (in der auch das Attribut *DroidfootActivity.worldClass* festgelegt werden muss) auf einfache Art und Weise möglich, Anpassungen des Android-Greenfoot-Players vorzunehmen. Hierzu sind bestimmte Attribute entsprechend zu setzen. Wenn Sie wie in Kapitel 5.1 beschrieben das Projekt *DFTemplate* als Ausgangslage für Ihr Projekt genutzt haben, finden Sie folgende Anweisungen in der Activity-Datei:

```
// help item available in the menu (true/false)?
Settings.help = true;

// speed item and speed slider available in the action bar (true/false)?
Settings.speed = true;

// background color of the droidfoot activity view
Settings.backgroundColor = Color.rgb(207, 241, 116);

// scale option available in the settings (true/false)?
Settings.scale = true;
// if true the world will always cover the whole screen
Settings.scaleDefault = true;

// orientation option available in the settings (true/false)?
Settings.orientation = true;
// if LANDSCAPE always landscape modus
// if PORTRAIT always portrait modus
// if SWITCH automatic change between portrait and landscape modus due
// to device rotations
Settings.orientationDefault = Settings.SWITCH; // LANDSCAPE, PORTRAIT, SWITCH

// username option available in the settings (true/false)?
Settings.username = true;
// name for the highscore table
Settings.usernameDefault = "Default";
```

Folgende Anpassungen sind möglich:

- **Hilfe**: Soll in der ActorBar bzw. im Overflow-Menü ein Hilfe-Symbol bzw. -Item erscheinen (Attribut *Settings.help*)?
- **Geschwindigkeitsregler**: Soll in der ActorBar das Geschwindigkeitssymbol bzw. der Geschwindigkeitsregler erscheinen (Attribut *Settings.speed*)? Wenn die Benutzer die Geschwindigkeit nicht ändern können sollen, muss dem Attribut der Wert *false* zugewiesen werden.
- **Hintergrundfarbe**: Ändern Sie die Hintergrundfarbe des Views, auf dem die Welt gezeichnet wird (Attribut *Settings.backgroundColor*).
- **Skalierung**: Soll im Einstellungen-Menü der Eintrag „Skalierung“ angezeigt werden (Attribut *Settings.scale*)?
- **Standardwert für Skalierung**: Hierüber können Sie angeben, ob die Welt immer den gesamten Bildschirm überdecken soll (Attribut *Settings.scaleDefault*)?

- **Bildschirmausrichtung**: Soll im Einstellungen-Menü der Eintrag „Bildschirmausrichtung“ angezeigt werden (Attribut *Settings.orientation*)?
- **Standardwert für Bildschirmausrichtung**: Hiermit können Sie den Standardwert für die Bildschirmausrichtung ändern zwischen immer Landscape, immer Portrait oder automatischer Wechsel zwischen Landscape und Portrait (je nach Lage des Android-Gerätes) (Attribut *Settings.orientationDefault*)?
- **Benutzername**: Soll im Einstellungen-Menü der Eintrag „Name“ angezeigt werden (Attribut *Settings.username*)?
- **Standardwert für Benutzernamen**: Hiermit können Sie den Standard-Benutzernamen ändern (Attribut *Settings.usernameDefault*)?

5.5 Weitergehende Anpassungen

Weitergehende Anpassungen sind durch entsprechende Änderungen im Projekt *Droidfoot* natürlich auch möglich. Anzumerken ist, dass die Implementierung von Droidfoot auch in dieser zweiten Version 2.0 im softwaretechnischen Sinne noch nicht besonders „sauber“ ist. Das wird mit einer der nächsten Versionen geändert. Droidfoot wurde aber umfangreich getestet und sollte keine größeren Fehler mehr enthalten.

6 Kontakt

Bei Fragen, Problemen oder Verbesserungsvorschlägen können Sie gerne Kontakt mit mir aufnehmen: Dietrich Boles, Universität Oldenburg, Email: boles@informatik.uni-oldenburg.de