Developerhandbuch

Armin Bernstetter, Stefan Ernst, Nicolas Fella

7. Dezember 2016

Zusammenfassung

Inhaltsverzeichnis

1 Einleitung/Aufgabenstellung **2**

2 Projektaufbau **2**
2.1 libGDX . . . 2
2.2 Gradle . . . 2
2.3 Getting started . . . 2
2.4 Ordnerstruktur . . . 2

3 Code **2**
3.1 Klasse Battletanks . . . 2
3.2 Package Screens . . . 2
3.2.1 MenuScreen . . . 2
3.2.2 GameScreen . . . 2
3.2.3 EndScreen . . . 2
3.3 Package Entity . . . 2
3.3.1 Klasse Entity . . . 2
3.3.2 Klasse Player . . . 2
3.3.3 Klasse Bullet . . . 2
3.3.4 Klasse Obstacle . . . 2
3.3.5 Enum Direction . . . 2
3.3.6 Enum Tanks . . . 2
3.4 Package Utility . . . 3
3.4.1 Klasse FileChooser . . . 3
3.4.2 Klasse FileListItem . . . 3

4 Dateien/Assets **3**
4.1 Die Maps . . . 3
4.2 Textures/Grafiken . . . 3
4.3 Skin/Fonts . . . 3
4.4 Sounds . . . 4

5 Diagramme **4**
5.1 Architekturdiagramm . . . 4
5.2 Aktivitätsdiagramm(e) . . . 4
5.3 Sequenzdiagramm(e) . . . 4

1 Einleitung/Aufgabenstellung

2 Projektaufbau

2.1 libGDX

2.2 Gradle

2.3 Getting started

2.4 Projektstruktur

3 Code

3.1 Klasse Battletanks

Die Klasse BattleTanks ist die Hauptklasse. Sie erbt von der LibGDX Klasse Game und verwaltet die Screens, den TextureAtlas sowie die Preferences. [...]

3.2 Package Screens

Alle Klassen in diesem Package implementieren das LibGDX interface Screen.

3.2.1 MenuScreen

Der MenuScreen besitzt zwei Stages, auf denen das Menü dargestellt wird. Die Hintergrundstage beinhaltet nur das Hintergrundbild, welches an die Fenstergröße angepasst und bei Bedarf gestreckt/gestaucht wird. Das gesamte Menü wird durch die zweite Stage dargestellt, die eine Table mainTable beinhaltet, auf der sich alle UI Elemente befinden. Diese werden jeweils von Methoden erstellt, die nach dem Schema 'createUIElement()' benannt sind und werden anschließend von der Methode 'create()' auf der mainTable angeordnet.

Zur Darstellung aller UI Elemente wird der im Ordner assets/data befindliche Skin uiskin.json verwendet. (Siehe 4.3 Skins/Fonts)

3.2.2 GameScreen

Im GameScreen befindet sich die gesamte Spiellogik. [...]

3.2.3 EndScreen

Der EndScreen erscheint sobald die eingegebene Zeit abgelaufen ist. Er ist ähnlich aufgebaut wie der MenuScreen und bekommt eine Liste an Playern übergeben. Diese werden mithilfe des internen PlayerComparator zuerst absteigend nach Kills, aufsteigend nach Deaths und danach aufsteigend nach PlayerNumber sortiert und in Form eines Scoreboards dargestellt. Zudem besitzt EndScreen einen Button, mit dem ein neues Spiel gestartet werden kann.

3.3 Package Entity

3.3.1 Klasse Entity

3.3.2 Klasse Player

3.3.3 Klasse Bullet

3.3.4 Klasse Obstacle

3.3.5 Enum Direction

3.3.6 Enum Tanks

Das Enum Tanks erstellt die fünf Spielfiguren. Jede davon besitzt einen Damage-, Armor- und HealthPoints-Wert sowie eine ReloadTime/Schussfrequenz. [...]

3.4 Package Utility

3.4.1 Klasse FileChooser

3.4.2 Klasse FileListItem

4 Dateien/Assets

4.1 Die Maps

Die verwendeten, bzw potenziell einlesbaren Maps sind .tmx Dateien, die unter anderem mit TiledMapEditor [INSERT LINK] erstellt werden können.

4.2 Textures/Grafiken

Der Hintergrund stammt von [INSERT LINK] Sämtliche im Game verwendeten Textures stammen von kenney.nl [INSERT LINK]. Dies beinhaltet die Textures aus denen die Map besteht, die Bullets sowie Panzer.

4.3 Skin/Fonts

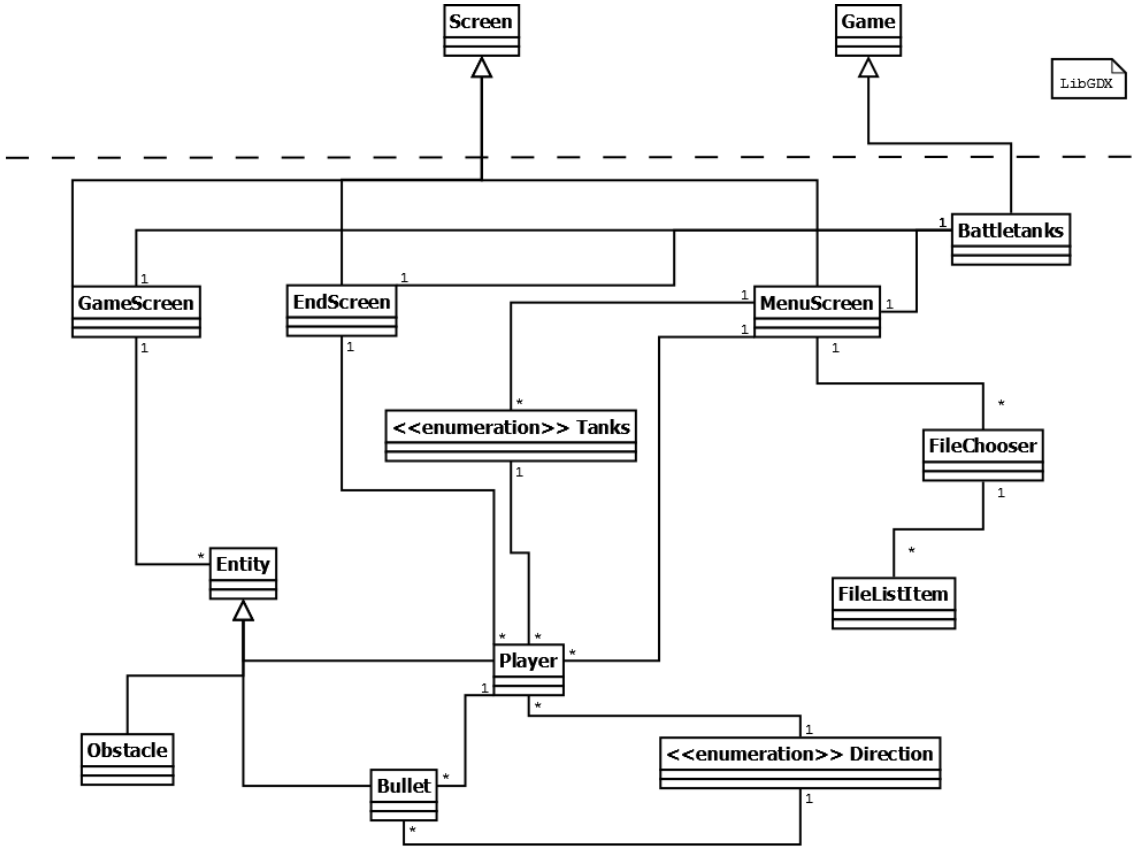
Der verwendete Skin sowie alle zugehörigen Dateien befinden sich im Ordner assets/data im core Projekt. Ausgenommen des im Menü und EndScreen verwendeten Fonts pixel.fnt stammen die verwendeten Dateien aus dem LibGDX Tests Repository [INSERT LINK] und werden in der LibGDX Dokumentation als default Dateien empfohlen. Zur näheren Information über die Funktionsweise von Skins siehe [LINK ZU LIBGDX WIKI].

Der Font pixel.fnt stammt von kenney.nl und wurde mithilfe des Programms Hiero von einem TrueTypeFont in einen BitMapFont umgewandelt.

4.4 Sounds

5 Diagramme

5.1 Architekturdiagramm



5.2 Aktivitätsdiagramm(e)

5.3 Sequenzdiagramm(e)