Ali Sina Iklil
01505118

Dokumentation - Computer Science Runner

Hier findet man eine Dokumentation der wichtigsten Spielmechaniken und eine technische Beschreibung des Projekts.

Spielmechaniken

ComputerScienceRunner ist ein Jump 'n' Run Spiel, bei dem man den Studienplan des Studiums Bachelor Informatik und Master Informatik durchspielen kann.

Spieler

Der Spieler kann einen männlichen oder weiblichen Charakter wählen, wenn er das Spiel zum ersten Mal spielt. Man kann den Charakter in den Optionen wieder ändern.

Levels

Es gibt insgesamt 10 Level im Spiel und jedes Level stellt sozusagen ein Semester dar. Am Anfang kann der Spieler nur das erste Level spielen und muss jedes Level freischalten, indem er das vorherige Level absolviert. Man kann jedoch in den Optionen auch alle Levels sofort freischalten. Dieses Feature habe ich deswegen eingebaut, da manche Spieler eventuell ein Level skippen möchten oder erst beim Master, also bei Level 7 beginnen wollen. Man kann auch den bisherigen Fortschritt in den Optionen löschen und wieder bei Level 1 beginnen. Jedes Level hat eine eigene Musik (siehe Credits) und ein eigenen Stil (z.B. Wüste, Eis, Grün, Herbst usw.). Die Musik und die Soundeffekt dürfen natürlich verwendet werden, solange Autor, ein Link zur Quelle und ein Link zur Lizenz bzw. zu den Lizenzen angegeben werden.

ECTS

In jedem Level bzw. Semester gibt es 30 ECTS zu sammeln. Die ECTS haben für den Spieler an sich keinen Nutzen, sondern sind nur ein Collectible wie in vielen Jump 'n' Runs. Der Spieler muss also nicht alle 30 ECTS haben um ein Level zu absolvieren. Es gibt auch gelbe ECTSBricks. Wenn der Spieler dagegen springt, kann der Spieler den ECTS Punkt, der üben dem ECTSBrick erscheint, einsammeln.

Coins

Es gibt im Spiel auch Coins, die der Spieler einsammeln kann. Diese haben so wie ECTS keinen Nutzen. Es gibt auch gelbe CoinBricks, welche optisch genauso aussehen wie ECTSBricks. Wenn der Spieler dagegen springt, kann der Spieler den Coin einsammeln.

Herzen

Der Spieler hat 3 Herzen. Wenn er 0 Herzen hat, gibt es ein Game Over. Man kann im Spiel natürlich auch Herzen einsammeln. Herzen verliert man wenn man einen Bug (also Käfer) an den Seiten oder unten berührt oder wenn man Lava oder Spikes (also Stacheln) berührt. Es gibt auch gelbe HeartBricks, welche optisch genauso aussehen wie ECTSBricks oder CoinBricks. Wenn der Spieler dagegen springt, kann der Spieler das Herz, welches über dem HeartBrick erscheint, einsammeln.

Informationskacheln bzw. InfoBricks
In jedem Level findet man Informationskacheln bzw. InfoBricks, gegen die der Spieler mit dem Kopf springen kann um den Text zu lesen. Dabei werden dem Spieler Inhalte vermittelt über die Lehrveranstaltungen die er besuchen muss bzw. kann. Man kann die Texte in "InfoBrickTexte.pdf" bzw. in "InfoBrickTexte .odt" nachlesen.

Fragen
Am Ende jedes Levels, werden dem Spieler 3 Fragen gestellt, jedoch gibt es beim Falschbeantworten keinen Nachteil für ihn, selbst wenn er alle Fragen falsch beantwortet. Man kann die Fragen in "Fragen.pdf" bzw. in "Fragen.odt" nachlesen.

Gegner
Im Spiel gibt es Bugs bzw. Käfer die als Gegner auftreten. Der Spieler kann von oben auf einen Käfer springen um ihn zu "zerstampfen". Es gibt insgeamt 3 Gegnertypen. Es gibt einen einfachen Bug, der nur von links nach rechts geht. Es gibt auch FlyingBugs, die fliegen. Es gibt SpearBugs, die Speere auf den Spieler werfen. Man kann auch auf die Speere springen und diese so zerstören. (Es gab in einer früheren Version auch JumpingBugs, jedoch gab es auf manchen Systemen Abstürze kurz nachdem ein JumpingBug "zerstampft" wurde. Ich konnte den Fehler leider nicht lösen und entschied mich daher, alle JumpingBugs aus dem Spiel zu entfernen.)

Technische Beschreibung

Das Spiel wurde mit Java geschrieben. Dabei wurde das libGDX Framework verwendet, welches relativ bekannt ist und platformübergreifende Spieleentwicklung ermöglicht. Als IDE habe ich Android Studio verwendet. Eigentlich wollte ich das Spiel für Android veröffentlichen, jedoch gab es einige Probleme, die beim Testen am PC nicht auftraten. Zum Beispiel stürzte das Spiel an der selben Stelle ab oder bei beweglichen Platformen blieb der Spieler stecken oder rutsche zur Seite ab. Diese Probleme gab es leider bei mehreren Smartphones, jedoch hatte ich beim Testen am PC keine dieser Probleme. Außerdem war die Steuerung am Smartphone zu ungenau für das Spiel, auch nachdem ich die Buttons vergrößert habe. Ich halte es daher für besser, das Spiel als einfache .exe Datei für den PC zu veröffentlichen, die man sich herunterladen kann.

Java Klassen

Hud.java: Zeigt Anzahl der Herzen, ECTS, Coins und das jetzige Semester im Spiel an
InfoWidget.java: Ist das Fenster mit dem Text, wenn der Spieler gegen ein InfoBrick springt

Brick.java: Ein einfacher Brick aus Holz, der nichts macht, wenn der Spieler dagegen springt
Coin.java: Kann der Spieler einsammeln
CoinBrick.java: Wenn der Spieler dagegenspringt, erscheint ein Coin, was man einsammeln kann
ECTS.java: Kann der Spieler einsammeln
ECTSBrick.java: Wenn der Spieler dagegenspringt, erscheint ein ECTS, was man einsammeln kann
Heart.java: Kann der Spieler einsammeln um ein Herz mehr zu haben
HeartBrick.java: Wenn der Spieler dagegenspringt, erscheint ein Herz, was man einsammeln kann
HorizontalPlatform.java: Bewegliche Platform die abwechselnd von links nach rechts geht
VerticalPlatform.java: Bewegliche Platform die abwechselnd von unten nach oben geht
InfoBrick.java: Wenn der Spieler dagegenspringt, erscheint ein Text (über das InfoWidget)
Trampoline.java: Hilft dem Spieler höher zu springen
InteractiveObject.java: Viele Klassen erben von dieser Klasse. Definiert Körper für Box2D

FlyingBug.java: Fliegender Käfer
Player.java: Klasse für Spieler. Steuerung des Spielers ist hier implementiert
Spear.java: Klasse für Speere, welche vom SpearBug geworfen werden können
SpearBug.java: Käfer, der Speere wirft
WalkingBug.java: Käfer, der nur abwechselnd von links nach rechts geht

CreditsScreen.java: Hier stehen die Credits für die Musik und die Soundeffekte
GameOverScreen.java: Wird angezeigt, wenn der Spieler stirbt
GenderSelectionScreen.java: Spieler kann hier männlichen oder weiblichen Charakter wählen
LevelSelectionScreen.java: Hier kann man das Level auswählen, welches man spielen will
MainMenuScreen.java: Hauptmenü des Spiels
OptionsScreen.java: Man kann Spielstand löschen, Charakter ändern oder alle Level freischalten
PauseScreen.java: Wenn der Spieler während dem Spiel auf den Pause-Button drückt
PlayScreen.java: Der Screen der das eigentliche Spiel darstellt
QuestionScreen.java: Hier werden die Fragen gestellt am Ende des Levels
SemesterCompletedScreen.java: Wird angezeigt, nachdem das Level und die Fragen fertig sind

ComputerScienceRunner.java: Hauptklasse des Spiels, wo sich globale Variablen befinden usw.
CustomOrthogonalTiledMapRenderer.java: Wird benötigt, da ECTS und Coins sich nicht weiterdrehen sollen, wenn der Spieler Text aus einem InfoBrick ließt
Question.java: Einfache Klasse, die eine Frage mit den möglichen Antworten darstellt
SmartphoneController.java: Wird nur benötigt für den Pause-Button im HUD
WorldContactListener.java: Wird aufgerufen, wenn es im Spiel zwischen zwei Box2D Körper einen Kontakt gibt. Wenn z.B. der Spieler mit einem ECTS kollidiert, dann verschwindet es und der ECTS Counter wird um eins erhöht

Tiled

Die Levels für das Spiel wurden alle mit dem Programm Tiled erstellt. Der Tiled Map Editor ist extrem bekannt und wird für Platformer aber auch für Top-Down-RPGs häufig verwendet. Für jedes Level gibt es eine eigene .tmx Datei die mit Tiled erstellt wurde. In diesen Dateien wird das gesamte Level definiert. Also wo sich Gegner befinden, welche Texte in den Informationskacheln stehen, welche Fragen am Ende des Levels gestellt werden, welche beweglichen Platformen im Level sind und so weiter. Man findet die Levels unter "android\assets\levels". Alle Assets (also Grafiken, Musik, Levels usw.) befinden sich unter "android\assets\".

Tiled Layer Erklärung

In Tiled gibt es Kachelebenen, wo die Grafik des Levels definiert wird und Objektebenen, wo einfach gesagt Rechtecke definiert werden für die Kollisionserkennung und z.B. für die Positionen der Gegner und der beweglichen Platformen und so weiter. Man sollte die Reihenfolge der Ebenen nicht ändern, da im Spiel über den Index der Layer zugegriffen wird und man Probleme bekommen würde. Wenn man einen neue Ebene (egal ob Objekt- oder Kachelebene) hinzufügt, sollte diese daher immer ganz oben sein.

Objektebenen:

Skybox: "Decke" des Levels, damit der Spieler nicht eventuell durch ein JumpingBug zu weit nach oben fliegt
VerticalPlatform: Für vertikale Platformen
HorizontalPlatform: Für horizontale Platformen
Trampoline: Rechteck dort machen wo man ein Trampoline haben will
Spikes(+Lava): Rechteck dort machen wo man Spikes oder Lava haben will
SpearBug: Rechteck dort machen wo man ein SpearBug haben will
FlyingBug: Rechteck dort machen wo man ein FlyingBug haben will
Bug: Rechteck dort machen wo man ein einfachen Bug haben will
OnewayPlatform: Wird benötigt wenn man wo drauf springen kann von unten
Goal: "Fahnenstange" am Ende der Levels. Hier stehen auch die Fragen und Anworten drinnen
CoinBricks: Rechteck dort machen wo man ein CoinBrick haben will
Coins: Rechteck dort machen wo man ein Coin haben will. Immer einzeln pro Coin machen
Wall: Damit der Spieler links und rechts nicht aus dem Level raus kann
InfoBricks: Rechteck dorthin wo man ein InfoBrick haben will. Text in Eigenschaften reinschreiben
HeartBricks: Rechteck dort machen wo man ein HeartBrick haben will
Hearts: Rechteck dort machen wo man ein Herz haben will
ECTSBricks: Rechteck dort machen wo man ein ECTSBrick haben will
ECTS: Rechteck dort machen wo man ein ECTS haben will. Immer einzeln pro ECTS machen
Bricks: Rechteck dort machen wo man ein normales Brick (was nichts bringt) haben will
Ground: Für den Boden, aber auch für starre Platformen oder z.B. Wände eines Raums usw.

Kachelebenen:

Graphics: Für alles andere außer dem Hintergrund. Falls man irgendwo ein ECTSBrick, CoinBrick oder HeartBrick hinsetzt, darf das Feld darüber nicht leer sein im GraphicsLayer, sonst crasht das Spiel, wenn der Spieler gegen das ECTSBrick, CoinBrick oder HeartBrick springt
Background: Nur für den Hintergrund des Spiels. Hintergrund ist immer einfärbig momentan

Sonstiges:

Bei Buttons gibt es leider keine Zeichen für "üäö" und "ÜÄÖ". Ich habe es aber so geändert, dass man andere Symbole verwenden kann, um diese Buchstaben darzustellen.

für ü
% für Ü

@ für ä
& für Ä

> für ö
© für Ö (Alt gedrückt halten und 0169 eingeben, dann erscheint dieses Symbol)

Verwendete Musik und Soundeffekte:

Musik:
https://opengameart.org/content/platformer-game-music-pack
https://opengameart.org/content/orchestral-adventure
https://opengameart.org/content/8bit-adventure
https://opengameart.org/content/8bit-style-music

https://opengameart.org/content/magic-space
(public domain)

Soundeffekte:
https://opengameart.org/content/platformer-jumping-sounds
https://opengameart.org/content/8-bit-platformer-sfx
https://opengameart.org/content/level-finish-fanfares
https://opengameart.org/content/win-sound-1
https://opengameart.org/content/bad-sound-2

https://opengameart.org/content/platformer-sounds-terminal-interaction-door-shots-bang-and-footsteps
(public domain)

https://opengameart.org/content/game-over-soundold-school
(public domain)