Pflichtenheft

Labyrinth
05.11.2015

1 Management- und Dokumentationsattribute

Dokumentationsattribute	
Autor(en)	Matthias Ridder, Tom Quinders, Lara Sievers, Jaqueline Timmermann
Eindeutige Teamnummer	D02
Namen der Teammitglieder	Matthias Ridder, Tom Quinders, Lara Sievers, Jaqueline Timmermann
Quelle	Idee des Projektteams, Praktikumsvorgaben
Version	0.1
Bearbeitungsstatus	Entwurf

2 Kurzfassung

Wir wollen ein Spiel entwickeln, in dem sich der Spieler in ↗Ego-Perspektive durch ein dreidimensionales Labyrinth bewegen muss. Ziel ist es den Ausgang des Labyrinths zu erreichen. Das Labyrinth wird aus einem zweidimensionalen Bild generiert.

3 Visionen und Ziele

/PZ10/ Es muss ein 3D-Labyrinth generiert werden
/PZ20/ Das Labyrinth wird vom Spieler in der ↗Ego-Perspektive durchlaufen

4 Rahmenbedingungen

/PR10/ Das Spiel soll auf einem Computer unabhängig vom Betriebssystem laufen
/PR20/ Als Programmiersprache soll Java mit ↗JOGL 2.3 verwendet werden
/PR30/ Das Spiel soll mit Maus und Tastatur bedienbar sein
/PR40/ Zielgruppe: Menschen im Alter von 8-99 Jahren, die Spaß an Gelegenheitsspielen in Ego-Perspektive haben

5 Kontext

/PK10/ Das Spiel soll auf einem herkömmlichen PC ausführbar sein
/PK20/ Das Spiel wird mit einer herkömmlichen Maus und Tastatur oder vergleichbaren Eingabegeräten bedient

/PK30/ Das Spiel wird in der IDE ↗Eclipse programmiert
/PK40/ Das Versionsmanagement wird über ↗Github realisiert

6 Funktionale Anforderungen

/PF10/ Es muss ein lauffähiges Programm entstehen
/PF20/ Der Spieler muss das Spiel aus der ↗Ego-Perspektive steuern
/PF30/ Es muss ein 3D-Labyrinths aus einem 2D-Bild generiert werden
/PF40/ Die 2D-Bilder müssen in Schwarz-Weiß erstellt werden, wobei schwarze Pixel Wände darstellen und weiße Pixel den begehbaren Pfad
/PF50/ Das Spiel muss einen blauen (0,0,255) Pixel als Start- und einen grünen (0,255,0) Pixel als Endpunkt erkennen können
/PF60/ Das Spiel kann rote (255,0,0) Pixel als sammelbare Gegenstände erkennen
/PF70/ Das Spiel muss eine Kollisionserkennung beinhalten
/PF80/ Das Spiel muss den Labyrinthausgang deutlich erkennbar darstellen
/PF90/ Der Spieler kann eigene 2D Bilder ins Spiel laden, um eigene Level zu erstellen
/PF100/ Es müssen Mappings auf die Oberflächen geladen werden
/PF110/ Das Spiel kann prozedural generierte Karten erstellen
/PF120/ Das Spiel kann eine Zeitmessung beinhalten
/PF130/ Das Spiel kann sammelbare Gegenstände enthalten, die dem Spieler bei Abschluss des Levels zusätzliche Punkte einbringen
/PF140/ Wenn das Spiel eine Zeitmessung oder einen Punktestand enthält, dann kann es einen ↗Highscore geben
/PF150/ Wenn das Spiel einen ↗Highscore beinhaltet, dann muss dieser lokal auf der Festplatte gespeichert werden
/PF160/ Das Spiel kann ein Hilfedokument zur Verfügung stellen
/PF170/ Das Spiel kann im Vollbildmodus ausgeführt werden
/PF180/ Das Spiel muss 4 voreingestellte Level beinhalten

7 Qualitätsanforderungen

Systemqualität	Sehr gut	Gut	Normal	Nicht relevant
Funktionalität		X		
Zuverlässigkeit		X		
Benutzbarkeit	X			
Effizienz			X	
Wartbarkeit			X	
Portabilität				X

Tabelle 1: Qualitätsanforderungen

8 Abnahmekriterien

- Funktionsfähige Kollisionserkennung
- Korrektes Einladen von Texturen
- Korrekte Levelerstellung aus dem 2D-Bild
- Gewinn des Spiels durch Erreichen des Zielpunktes

9 Subsystemstruktur (optional)

10 Glossar

- Ego-Perspektive: Eine in Spielen übliche Perspektive, die dem Spieler die Sicht aus der Perspektive der Spielfigur ermöglicht
- JOGL: Java Bindings for OpenGL - Eine OpenGL Anbindung für Java
- Eclipse: Eine Entwicklungsumgebung zur Entwicklung von Java-Programmen
- Github: Ein Programm zur Versionsverwaltung
- Highscore: Eine Rangliste, in der die besten Spieler aufgeführt werden

11 Referenzen

11.1 Hinweis zu dieser Vorlage

Die Vorlage für dieses Pflichtenheft wurde Balzert (2009), S. 492 ff. entnommen.

11.2 Referenzen

Balzert, Helmut (2009). Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering. 3. Auflage. Heidelberg: Spektrum, Seite 492 ff.