Softwaredokumentation Tetris

Anforderung & Beschreibung des Programmes (Für User)

Tetris ist ein klassisches Rätselspiel, bei dem die Spieler zufällig generierte geometrische Blöcke (genannt Tetrominoes) in einer Matrix arrangieren müssen. Jeder Block fällt von oben nach unten und der Spieler muss ihn drehen und positionieren, um Linien zu bilden. Wenn eine vollständige horizontale Linie ohne Lücken erstellt wird, verschwindet sie, und der Spieler erhält Punkte. Das Spiel wird komplizierter, da die Geschwindigkeit der fallenden Blöcke zunimmt. Das Spiel endet, wenn die Blöcke den oberen Rand des Spielfeldes erreichen.

Verwendete Technologien und Bibliotheken (Für Programmierer)

Dieses Tetris-Spiel wurde in Java entwickelt, unter Verwendung von Java Swing für das User Interface und Java AWT für die Grafik und die Farbverwaltung. Darüber hinaus wurden verschiedene Java-Klassen verwendet, um verschiedene Spielkomponenten wie Blöcke, Spiellogik und Kollisionserkennung zu verwalten.

- Block-Klasse: Diese Klasse repräsentiert einen Tetris-Block. Sie enthält Informationen über die Form, Farbe und Position des Blocks.

- Game-Klasse: Diese Klasse verwaltet den Spielzustand. Sie enthält Informationen über das Spielfeld, die aktuelle und nächste Blöcke, den Punktestand und den Highscore. Sie ist auch für die Initialisierung und Aktualisierung des Spiels verantwortlich.

- Gui-Klasse: Diese Klasse ist verantwortlich für das Erstellen und Aktualisieren der grafischen Benutzeroberfläche. Sie verwendet Swing-Komponenten, um das Spielfeld, die Blöcke und den Punktestand anzuzeigen.

- DrawGame, DrawInterface und DrawMenu-Klassen: Diese Klassen sind Unterklassen von JLabel und sind für das Zeichnen der verschiedenen Komponenten des Spiels verantwortlich.

- KeyHandler-Klasse: Diese Klasse ist verantwortlich für die Verarbeitung der Tastatureingaben des Benutzers.

- GameLoop-Klasse: Diese Klasse repräsentiert den Hauptspielzyklus. Sie ist für die Aktualisierung des Spiels in regelmäßigen Abständen verantwortlich.

Design & Bedienbarkeit

Das Spiel folgt einer minimalistischen Designphilosophie mit klar definierten Blöcken und Farben. Es gibt drei Hauptbildschirme - das Menü, das Spiel und die Spieloberfläche. Jeder Bildschirm wird von einer separaten Java-Klasse verwaltet. Die Benutzeroberfläche ist intuitiv und leicht zu bedienen. Spieler verwenden die Pfeiltasten, um die Blöcke zu bewegen die Leertaste um die Blöcke zu drehen, und die ESC-Taste, um das Spiel zu pausieren.

Das Design des Spiels folgt einer modularen Architektur, bei der jede Klasse eine spezifische Aufgabe hat. Dies erleichtert das Verständnis, die Wartung und die Erweiterung des Spiels.

Ordnungsgemäße Funktion

Das Spiel funktioniert wie erwartet und hält sich eng an die klassischen Regeln von Tetris. Es bietet auch zusätzliche Funktionen wie Punktezählung und Highscore-Tracking. Die Spiellogik ist solide und die Kollisionserkennung funktioniert einwandfrei.

Kostenschätzung anhand von User Storys & Planning Poker

Um die Kosten zu schätzen, verwenden ich User Stories und Planning Poker. User Stories sind einfache Beschreibungen der Funktionen, die ein Benutzer von der Software erwartet. Bei Planning Poker schätzt jedes Mitglied des Entwicklungsteams, wie viel Arbeit jede User Story erfordert.

User-Storie 1: Ich möchte die Blöcke drehen können, um sie besser platzieren zu können.	13
User-Storie 2: Ich möchte die Fallgeschwindigkeit beschleunigen können.	8
User-Storie 3: Ich möchte das Spiel jederzeit pausieren können.	5
User-Storie 4: Ich möchte meine Punktzahl sehen können.	5
User-Storie 5: Ich möchte den Highscore sehen können.	3
Total	34

Ich habe entschieden, die Stories mit den 5 Punkten als Grundlage zu nehmen und diese auf 2 Stunden zu schätzen.

$$\frac{2\text{ h} * 34\text{ Story Points}}{5\text{ Story Points}} = 13.6\text{ h}$$

Bei einem Stundenlohn von 140.- ergibt dies Fr. 1904.-

Diese Kostenschätzung umfasst nur die Entwicklung. Weitere Kosten könnten für das Design, die Projektleitung, das Testen, die Bereitstellung und die Wartung anfallen.

Testfälle

Testfall 1: Blockbewegung - Überprüfen, ob sich die Blöcke korrekt nach links, rechts und unten bewegen und ob sie korrekt drehen.

Testfall 2: Linienlöschung - Überprüfen, ob eine vollständige Linie korrekt gelöscht wird und ob die darüber liegenden Blöcke korrekt herunterfallen.

Testfall 3: Punkteberechnung - Überprüfen, ob die Punkte korrekt berechnet werden, wenn Linien gelöscht werden.

Testfall 4: Spiel Bedingung I - Überprüfen ob die Kollison zwischen den Blöcken und den Aussenrand richtig funktioniert.

Testfall 5: Spiel Bedingung II - Überprüfen ob die Kollison zwischen den Blöcken richtig programmiert ist, Kollision von oben, unten sowie von link und rechts zu den anderen Blöcken.

Testfall 6: Spielende Bedingung - Überprüfen, ob das Spiel endet, wenn die Blöcke den oberen Rand des Spielfeldes erreichen.

Testfall 7: Pausenfunktion - Überprüfen, ob das Spiel beim Drücken der ESC-Taste pausiert und beim erneuten Drücken fortgesetzt wird.

Testfall 8: Highscore-Anzeige - Überprüfen, ob der Highscore korrekt gespeichert und angezeigt wird.