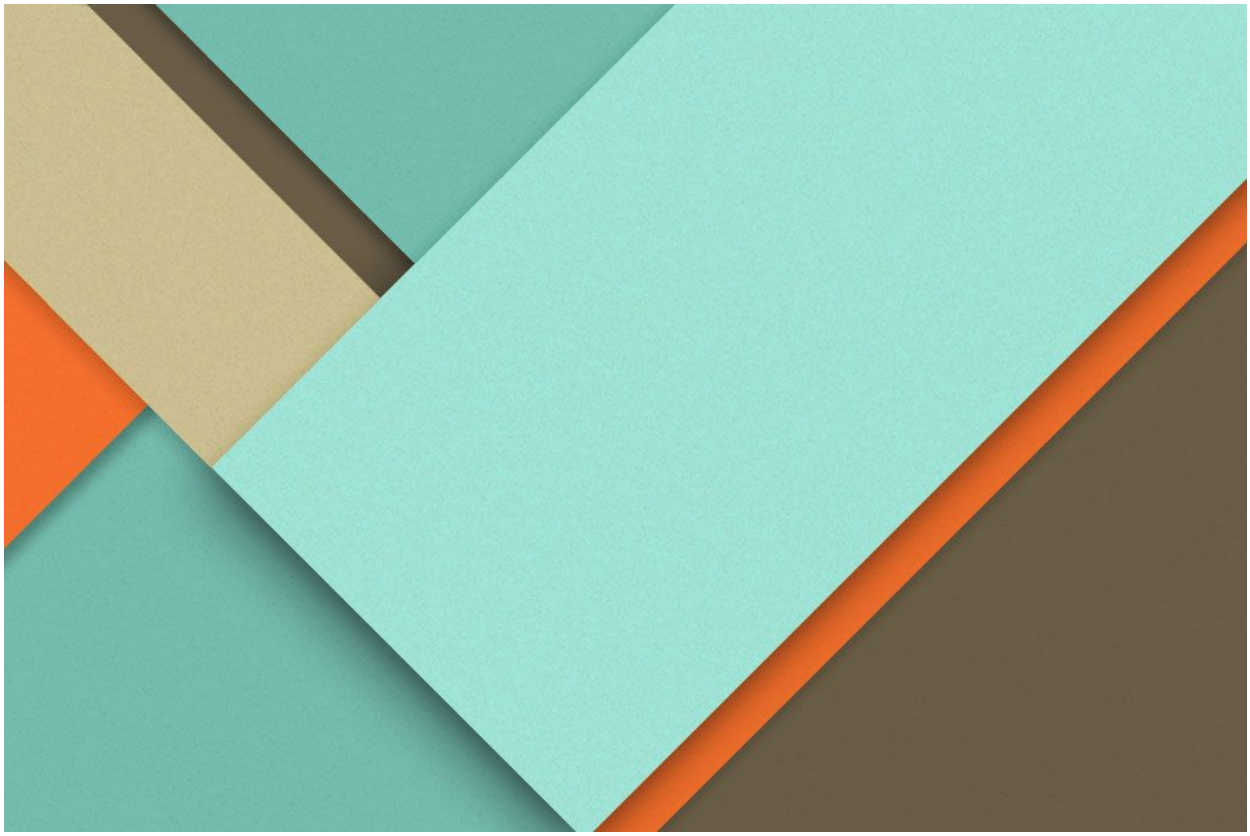


4 Gewinnt

Ausarbeitung Multimediaprogrammierung WS2018/19

Sven Alexander Weikert

10055756
weikert.svenalexander@fh-swf.de

Yasin Demirbas

10032931
demirbas.yasin@fh-swf.de

Inhaltsverzeichnis

1. Einleitung

4-Gewinnt

Der Spieleklassiker von 1974 wird auch 45 Jahre nach erscheinen noch mit Freude gespielt und hat bereits Einzug auf Bildschirme jedweder Art gefunden.

Die Regeln sind einfach, wer zuerst vier Spielsteine seiner Farbe horizontal, vertikal oder diagonal in einer Reihe platziert gewinnt das Spiel.

Projekthintergrund

Im 5. Fachsemester des Studiengangs Informatik B.Sc. an der Fachhochschule Südwestfalen befassen sich die Studenten unter Anderem mit dem Modul Multimediaprogrammierung.

Multimediprogrammierung behandelt die Implementierung von dreidimensionalen Szenerien unter Verwendung von OpenGL/ES und QML.

Ziel dieser Ausarbeitung

Diese Ausarbeitung beschreibt die Implementierung und den Umgang mit unserer Interpretation von 4-Gewinnt.

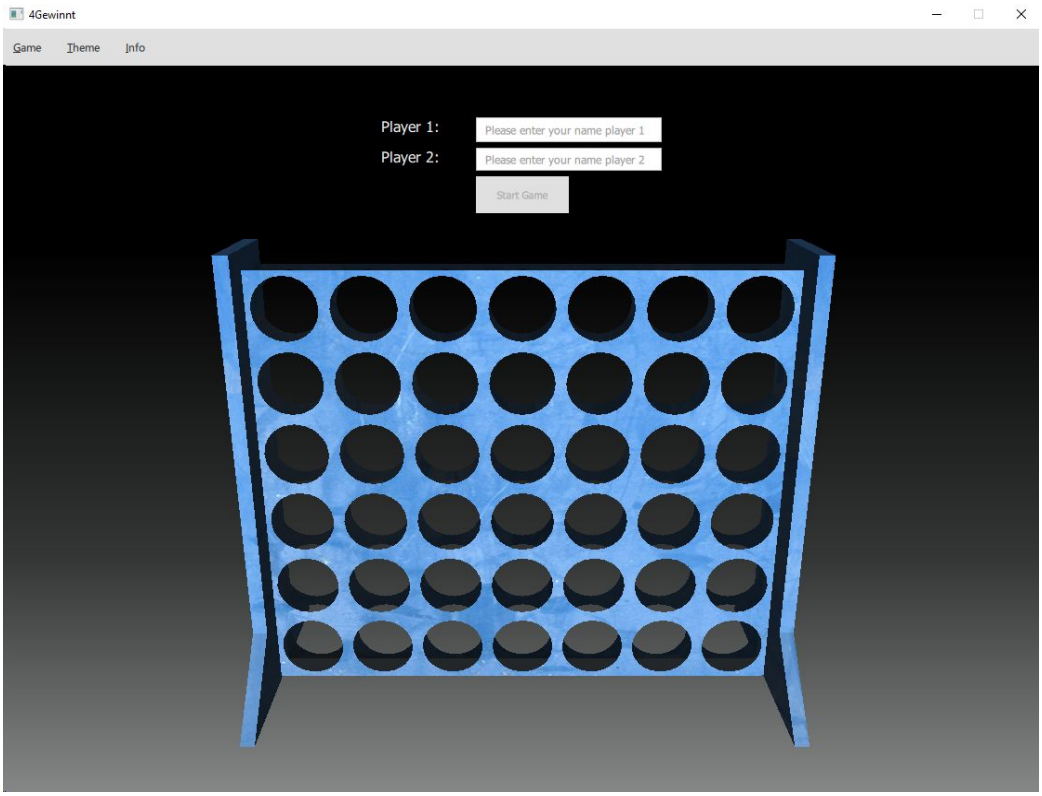
Neben einem User-Guide und einer Installationsanleitung werden wir den Entwicklungsprozess ausarbeiten und auf die Hauptbestandteile der Implementierung eingehen.

2. User-Guide

2.1. Spielbeginn

Nach dem Start der Anwendung ist das Spielgerüst und eine Eingabemaske zu sehen. Die Spieler geben jeweils ihren Namen ein und starten das Spiel mit dem Button “Start Game”.

Spielernamen bestehen aus Groß- und Kleinbuchstaben, haben mindestens drei und maximal 15 Zeichen. Sonderzeichen, Umlaute, Zahlen und Leerzeichen sind nicht zulässig.



2.2. Spielablauf

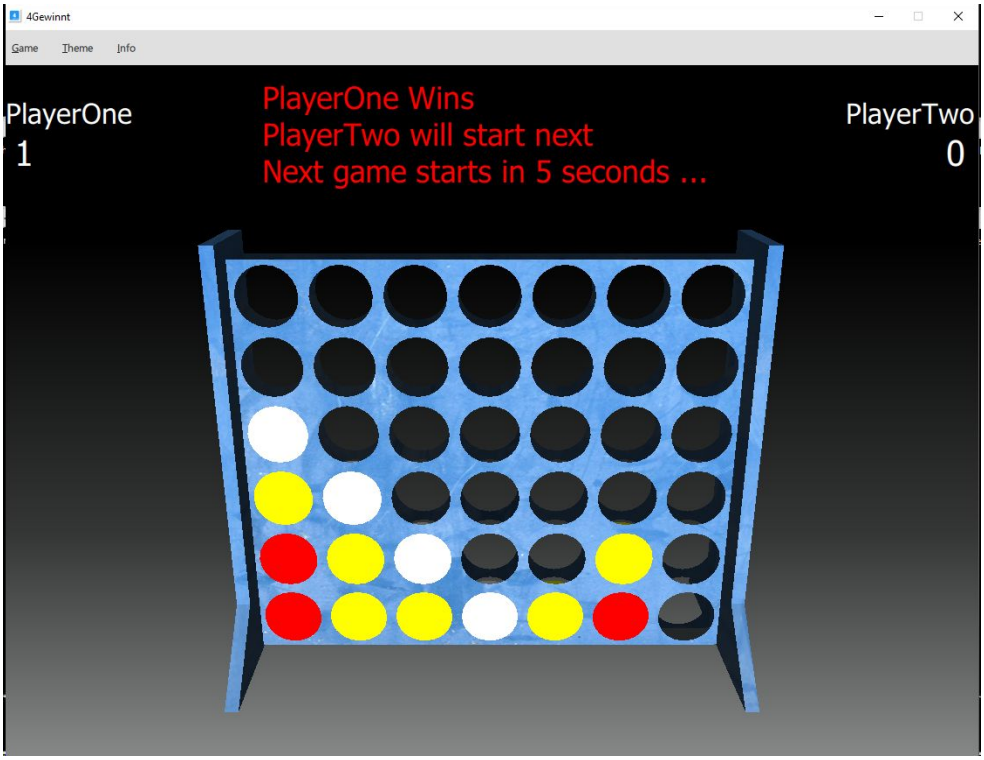
Mit betätigen des “Start Game” - Buttons startet die erste Runde und es wird eine Reihe von Buttons sichtbar über die eine Spielscheibe eingefügt werden kann.

Spieler 1 kann nun seine erste Spielscheibe platzieren. Daraufhin ist Spieler 2 am Zug.

2.3. Spielende

Sobald ein Spieler es schafft vier Spielscheiben horizontal, vertikal oder diagonal zu platzieren, werden die entsprechenden Steine weiß gefärbt.

Dem Sieger wird ein Punkt auf dem Scoreboard gutgeschrieben und der Verlierer darf die nächste Runde beginnen.



Gibt es nach 42 Spielscheiben keinen Sieger endet diese Runde Unentschieden und das Spiel wird automatisch neu gestartet.

2.4. Regeln

Jeder Spieler darf nur eine Scheibe pro Zug platzieren.

Sollte einer der Spieler unbeabsichtigt oder mit Vorsatz eine Scheibe der gegnerischen Farbe platzieren ist diese Runde ungültig.

Für diesen Fall kann die aktuelle Runde zurückgesetzt werden:

Menüleiste ⇒ Game → Restart

Es liegt in der Verantwortung der Spieler, dass diese Regel befolgt wird.

2.5. Features

Scoreboard

Der aktuelle Punktestand der Spieler wird unter dem Spielernamen angezeigt. Nach jeder Runde die einen Sieger hervorbringt wird dieser aktualisiert.

Wechsel der Spieler

Die aktuellen Spieler können auf Wunsch verändert werden. Beispielsweise kann bei mehr als zwei Spielern im Modus “Best of five” gespielt werden.

Menüleiste ⇒ Game → Change Players

Beim betätigen dieser Schaltfläche öffnet sich eine Warnung mit dem Hinweis, dass der aktuelle Spielprozess zurückgesetzt wird.

Mit dem Button “Continue game” kann das aktuelle Spiel fortgesetzt werden.

Der Button “I’m sure” setzt den Spielstand zurück und öffnet die Eingabemaske für die neuen Spieler.

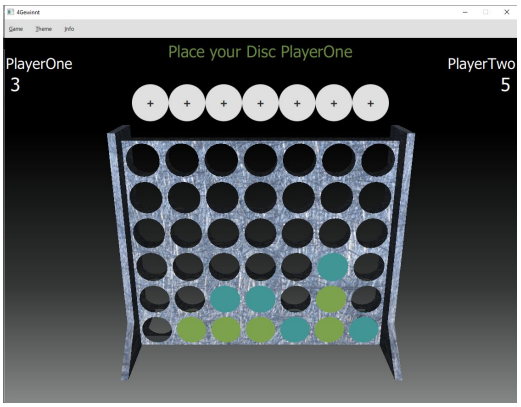
Wechsel des Layouts

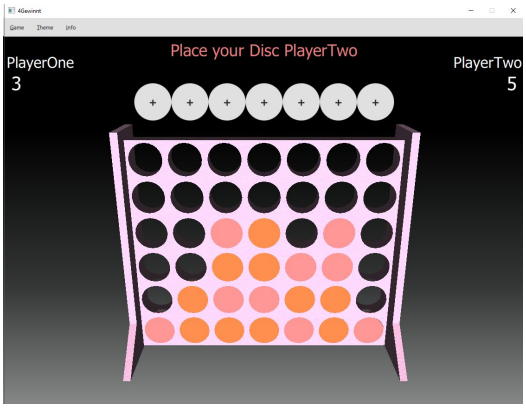
Das Spielfeld kann in drei Varianten dargestellt werden.

Die klassische Variante mit einem blauen Gerüst sowie roten und gelben Spielscheiben ist defaultmäßig eingestellt.

Zusätzlich bieten wir eine “versteinerte” Version an und auf Wunsch zahlreicher User ein exklusives “Princess-Theme” des Spiels.

Menüleiste ⇒ Theme → Classic || Stone || Princess





3. Technologiebeschreibung

3.1. Entwicklungs- & Anwendungsumgebung

Das gesamte Projekt wurde auf einem Windows 10 64-Bit-System entwickelt.

Die Android-Spezifikationen unterstützen die Ausführung ab Android 4.1. Getestet wurde die Anwendung während der Entwicklung auf Android 8.1.

3.2. Verwendete Bibliotheken

Implementiert wurde das Programm in C++ mit dem Anwendungsframework Qt und der OpenGL ES API.

Die Benutzungsoberfläche ist mit QML modelliert. QML erlaubt die Einbindung von JavaScript um z.B. Bindings zwischen Attributen von QML-Komponenten und C++-Attributen zu realisieren.

Des weiteren verwenden wir das von Professor Roth zur Verfügung gestellte Grundgerüst zur Darstellung einer 3D-Szenerie.

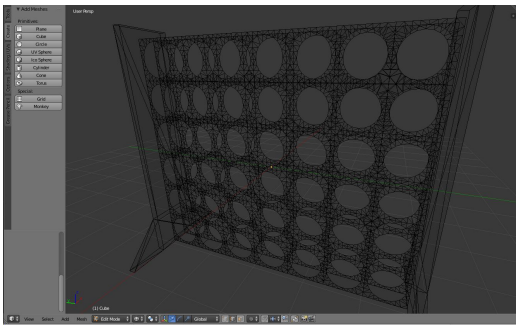
3.3. Weitere Entwicklungstools

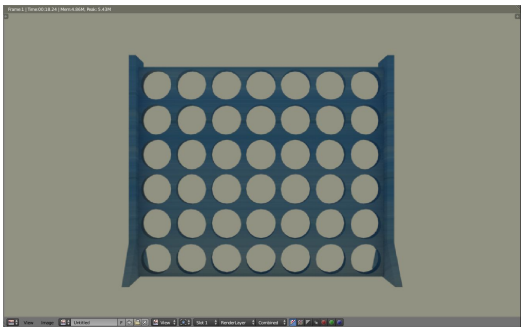
Blender

Blender ist eine 3D-Grafik-Suite zum modellieren, texturieren und animieren von dreidimensionalen Objekten.

Durch die Einführung von Prof. Roth in Blender, der Dokumentation und die Videos auf dem Youtube-Kanal Pitchfrog gelang es uns ein originalgetreues Modell nachzubilden.

Dieses modellierte Objekt wurde anschließend als Wavefront-Datei (.obj) exportiert.





Blenderkonverter

Dies ist ein von Eugen Iwanov entwickeltes Programm zur Konvertierung von Wavefront-Dateien in ein schnell ladbares Binärformat (.dat).

Ursprünglich wurde dieses Programm für das Enigma-Projekt von Prof. Roth entwickelt.

Das konvertierte Format unseres Spielgerüsts kann dadurch in den OpenGL-Kontext importiert und gerendert werden.

GitHub/GIT for Windows

Durch die GIT GUI gelingt es auch bash-scheuen Entwicklern ein Remote-Repository zu erstellen und zu verwalten.

Doxygen

Ein Software-Dokumentationswerkzeug zur Generierung von Dokumentationen in verschiedenen Ausgabeformaten.

Tabellarische Übersicht:

Software	Version
Betriebssystem	Windows 10 Education 64-Bit Android 8.1 (Oreo) API level 27
Compiler	MinGW 5.3.0 32-Bit Android ARMv7
Entwicklungsumgebung (IDE)	Qt Creator 4.8.1
Bibliotheken, API's & Frameworks	Qt 5.11.2 QtQuick 2.11 QtGraphicalEffects 1.0 OpenGL ES 2.0
Modellierung	Blender 2.79
Konvertierung	Blenderkonverter
Dokumentationsgenerierung	doxygen 1.8.15
Versionsverwaltung	GIT for Windows v2.20.1

4. Installationsanleitung

4.1. Windows

Entpacken des Zip-Archivs

Schritt 1) Kopieren Sie die Datei “4Gewinnt.zip” in das gewünschte Installationsverzeichnis.

Schritt 2) Klicken Sie mit der rechten Maustaste auf die Datei und wählen Sie den Menüpunkt “Alle Extrahieren...”.
Befolgen Sie die Anweisungen in dem Dialogfenster.

Schritt 3) Nach diesem Vorgang finden Sie eine ausführbare Datei namens “4Gewinnt.exe” in dem Verzeichnis.
Mit Rechtsklick → Senden an → Desktop können Sie eine Verknüpfung auf Ihrem Desktop erstellen.

Ausführen des Installers

Alternativ können Sie die Installation automatisiert ausführen lassen.

Schritt 1) Führen Sie die Datei “Setup4Gewinnt.exe” aus und befolgen Sie die Anweisungen in dem Dialogfenster.

Schritt 2) Öffnen Sie den Explorer und wechseln Sie in das Verzeichnis, das Sie bei der Installation angegeben haben.

Schritt 3) In dem Unterverzeichnis 4Gewinnt finden Sie die Datei “4Gewinnt.exe”.
Mit Rechtsklick → Senden an → Desktop können Sie eine Verknüpfung auf Ihrem Desktop erstellen.

4.2. Android

Schritt 1) Kopieren Sie die Datei “4Gewinnt.apk” in ein beliebiges Verzeichnis auf Ihrem Android-Device.

Schritt 2) Öffnen Sie den Dateimanager auf Ihrem Android-Device und wählen Sie in der linken Spalte “Installationsdateien” aus.

Schritt 3) Führen Sie die Datei “4Gewinnt.apk” aus und befolgen Sie die Anweisungen des Dialogfensters.
Anschließend finden Sie die App auf Ihrem Home-Bildschirm.

5. Übersichtsdokumentation über die erstellten Klassen

Vorbedingungen

Das Projekt basiert auf den von Prof. Roth zur Verfügung gestellten Klassen zur Darstellung einer dreidimensionalen Szenerie. Zu ihnen gehören:

- GLItem
- GLBody
- GLPoint
- GLESRenderer
- GLColorRgba
- ShaderDebugger

Projektspezifische Klassen

Das Spielfeld

Die Klasse *Board* ist ein statisches Raster von 7 x 6 Quadraten. Es wurden im Vorfeld die Koordinaten der drei Vektoren im Blendermodell abgelesen, welche ein Dreieck im Feld oben links ergeben.

Mit Hilfe von Deltas der verschiedenen Vektorkomponenten konnten wir Einheitsvektoren definieren, sodass das Raster in Trianglestrips zusammengesetzt werden kann. Weiterhin kann mit ihnen die zentrale Position eines Feldes bestimmt werden um eine Scheibe zu platzieren.

Gerendert wird das Spielfeld nur zum Debuggen und wurde im Quelltext auskommentiert.

Die Spielsteine

Die Klasse *GLDisc* wurde während des Praktikums entwickelts und für unser Konzept modifiziert.

Sie enthält ein Attribut zur Speicherung von Feldkoordinaten und zwei Attribute die benötigt werden um den Spielstein oben in das Gerüst einzuführen und bis an seine finale Position gleiten zu lassen.

Beim ersten Rendering eines Objekts wird das Attribut m_isFirst auf seinen Wert geprüft. Dadurch wird die erste Verschiebung der Scheibe verhindert. Anschließend wird m_isFirst mit False belegt. Durch einen Timer kann ab dem nächsten Rendering die Scheibe Schrittweise in y-Richtung verschoben werden, solange bis das Attribut m_stepsToPosition den Wert 0 enthält.

Durch Feintuning an dem Timerinterval, der Schrittweite und der Berechnung der “Schritte bis zur finalen Position” konnte eine flüssige Animation erzeugt werden.

Das Spiel

Der Aufbau und die Logik des Spiels ist in der Klasse MyGlItem enthalten.

Das Gerüst wird in den OpenGL-Kontext geladen und das Spielfeld wird instanziiert.

Die Interaktionen auf der QML-Oberfläche werden in verschiedenen Slots verarbeitet. Beispielsweise das Einfügen einer Spielscheibe oder das Starten eines neuen Spiels.

Eingefügte Spielsteine werden in zwei Mengen unterteilt: rote Steine und gelbe Steine, unabhängig davon welche Farbe tatsächlich dargestellt wird.

Nach jedem eingefügten Stein wird zum einen geprüft ob horizontal, vertikal oder diagonal vier Steine in Reihe platziert sind. Ist dies der Fall wird die Reihenfolge für die nächste Runde festgelegt, der Punktestand aktualisiert und ein neues Spiel gestartet.
Zum anderen wird geprüft ob die eingefügte Spielscheibe die 42. war. Führt diese nicht zum Sieg wird automatisch ein neues Spiel gestartet.

Zur Synchronisierung von internem und dargestellten Spielablauf werden QProperties für den nächsten Spieler, die nächste Farbe, die nächste Reihenfolge und den nächsten Spielstand verwendet.

Zugehörige Signale werden emittiert wenn der Attributwert verändert wird. Entsprechende Eventhandler sind in den QML-Dateien mit JavaScript implementiert.

6. Fazit

Der Ablauf des Projekts kann mit einer Schnitzeljagd verglichen werden. Mit erreichen eines Meilensteins, wurde man direkt vor eine neue Herausforderung gestellt.

Insgesamt verlief die Implementierung und die Anfertigung dieser Ausarbeitung über einen Zeitraum von vier Wochen.

Alles in allem sind wir sehr zufrieden mit unserem Ergebnis und dem Feedback der bisherigen Tester.

7. Erklärung über die selbstständige Anfertigung

Hiermit erkläre ich Sven Alexander Weikert, dass ich die vorliegende Ausarbeitung selbständig erarbeitet und alle verwendeten Hilfsmittel angegeben habe.

_____

Datum, Unterschrift

Hiermit erkläre ich Yasin Demirbas, dass ich die vorliegende Ausarbeitung selbständig erarbeitet und alle verwendeten Hilfsmittel angegeben habe.

_____

Datum, Unterschrift

8. Literaturverzeichnis

Qt Reference Documentation

https://doc.qt.io/qt-5/reference-overview.html

Blender Reference Documentation

https://docs.blender.org/manual/en/latest/

Android Developer API Reference

https://developer.android.com/reference

Youtube-Channel “Pitchfrog”

https://www.youtube.com/channel/UCl52Y7Nqkstzuu0RWehbFKA/about