

Politechnika Warszawska

Wydział Elektroniki i Technik
Informacyjnych



Praca dyplomowa inżynierska

na kierunku Computer Science
na specjalności Computer Systems and Networks

Tworzenie aplikacji multiplatformowych na podstawie tworzenia silnika do
gier "match three"

numer pracy według wydziałowej ewidencji prac {liczba}

Krzysztof Rudnicki

numer albumu 307585

promotor
dr hab. inż. Tomasz Martyn

konsultacje
dr hab. inż. Tomasz Martyn

Warszawa 2023

Streszczenie pracy

Tworzenie aplikacji multiplatformowych na podstawie tworzenia silnika do gier "match three"
{SŁOWA KLUCZOWE}

Thesis abstract

Example of creation of multiplatform apps on basis of creating match-three game engine
{KEYWORDS}

Contents

1	Introduction	5
1.1	Choice of graphic rendering API	5
1.2	Choice of OpenGL Library	5

Chapter 1

Introduction

1.1 Choice of graphic rendering API

There are 3 main APIs for graphical rendering

- DirectX
- OpenGL
- Vulkan

DirectX developed by Microsoft focuses on Windows operating systems and Microsoft line of consoles Xbox, it is deemed as being harder with more low level programming and requiring better understanding of how underlying mechanisms work but in turn offers functionalities and better performance. It does not have free license.

OpenGL is developed by Khronos Group and offers good compatibility, especially if using OpenGL ES subset which works on Windows, Linux, Mac OS, Android, iOS and all major consoles. It is widely recognized as easiest of APIs and most popular choice for writing first game engine. On the other hand it lacks some of more advanced features which have to be written manually. It uses open source license similar to BSD

Vulkan is also developed by Khronos Group and as such is deemed as a spiritual successor of OpenGL with focus on using modern C++ features and fixing issues created by OpenGL 30 years old development time. Out of these three it is recognized as the hardest one as it is both complicated and newest. Similarly as OpenGL it uses open source license, namely Apache License 2.0.

Considering all of those characteristics I decided to go with OpenGL API, specifically OpenGL ES subset with its focus on compatibility as making a multiplatform application is one of the focuses of this thesis. I decided that since this will be my first attempt at game engine development I need something that is relatively easy and has a lot of resources online. I would most likely not use advanced features of Vulkan and DirectX and therefore finish my thesis before approaching problems where OpenGL does not deliver more complicated architecture. From my own private preferences I also prefer software with open source license.

1.2 Choice of OpenGL Library

There are 4 basic OpenGL libraries that I considered:

- freeGLUT
- SDL
- SFML
- GLFW

freeGLUT was created as opensource alternative to GLUT, is considered to be the worst out of all 4, written in archaic way, using C or very old C++, which in turn results in unexpected "buggy" behaviour, it is also not really popular with lack of online guides

SDL - Simple DirectMedia Layer has big userbase, it is not designed to be used as a standalone library and requires additional libraries to do networking or to create more complex applications.

SFML is the library with most features out of all 4, it supports networking, audio and has system features by default. It uses modern object oriented C++. Main problem with SFML is that it is not very popular API, therefore troubleshooting problems with SFML is quite hard and it has only few use guides online

GLFW is an library that is both the most popular and with fewest features by default. It forces users to use additional libraries for networking, sound, physic calculations and so on but in turn is also quite small and flexible. It has biggest community and a lot of guides, like one hosted at learnopengl.com or one created by programming youtuber Cherno

I decided to use GLFW library. I wanted something that is relatively easy to troubleshoot and has abundance of learning materials online.