OGRE

Release 0.0

Jun 01, 2020

Contents

OGRE - Open Games Room for Everyone, is a free players lounge/lobby system that devs can use for their Multi Players games.

Code and other contributions are welcome. The code is hosted on GitHub.

Contents:

CHAPTER 1

Introduction

1.1 In the beginning...

It all started with my youngest son picking up introductory computer science courses at high school. When the Covid-19 crisis and social distancing response started, I decided to give him some additional help with algorithmics and Python programming. As a learning exercise, we started a small project to implement a Connect Four game in Python. The initial version was a text-based player versus player implementation. From here, we used Pygame to go for a graphics and sound version, and then a player versus computer version using Minimax algorithm... (we'll publish it one day, once we tidy up the source code)

Getting interested in Python and Pygame, I began to write other games to follow up on this promising path. With John Horton Conway's untimely passing, I thus wrote a little game of life implementation as a tribute.

The next step was to write a multi-player game...

1.2 How to write a multi-player game with Pygame?

Full of hope, I went to the Pygame group on Facebook and asked if there was some framework to make multi-players network games with PyGame. To my surprise the answer seemed to be negative, people using relatively low level network APIs such as UDP Communication or python-socketio.

That was not what I was searching for!

Of course, a network API would enable two players, already knowing each other and having already decided on a playing time, to play some game, but I had more general things in mind.

I then headed to GitHub to look for games/players lounge/lobby software. I found some interesting projects (for example, with focus on matchmaking, blockchain usage or Linux OS) but none exactly fitting my purpose, with the right blend of simplicity and scalability.

So I decided to go for it myself.

1.3 The issues I want to address

Perhaps the first issue with multi-player games would be **finding other players**. In order to solve that one, you need some gaming platform which gathers players. Players will do that if it's the reference platform for their favourite game (ie. chess, poker, etc.) or if it has a wide enough range of games to choose from and players to play with.

Of course there are already widely successful gaming platforms such as Steam, or open source plaforms such as Lutris, but apparently none focusing on casual, non commercial, multi-OS/hardware, homemade games using tools such as Pygame.

Another issue, at least for a new gaming platform, would be **finding players available for a specific game at a specific time**. If there are not enough available players for an immediate game launch, planning a game for a later time and enabling players/teams to register would be needed. For example, for organizing friday's evening wargame session with your friends around the world.

The gaming platform should be **accessible both through a web site and from inside your multi-player games**. The later would require APIs and code samples to bootstrap game developers.

And last but not least, such a gaming platform would have to be **not controlled by a single person or company** (and thus be distributed) and be **respectful of the privacy of its players** (ie. not collecting or abusing personal information more than necessary). With complex embedded software such as web, database and message queue servers, these infrastructure parts would have to be packaged as readily available virtual machine images or Docker containers, for use in the Cloud, on dedicated servers or private machines. Game accounts could also be distributed and even rely extensively on social login in other platforms.

1.4 Perfect information... or not

Furthermore, the gaming platform would have to **enable both perfect information games and imperfect ones**.

In a perfect information game such as Connect Four or Chess, all the past and current state of a game is known to all the players. Thus it is possible to have Client to Client software interaction, each running the full state of the game and using the gaming platform as an input for the opponent's moves.

In an imperfect information game such as a cards game or a strategic simulation game, Client software would focus on displaying the known state of the game for each player and sending their moves to some central server holding the full state of the game, resolving moves and sending back the new known state to each player. To some extent, the Server part would be a kind of electronic arbitrator, limiting possibilities to cheat, though the trickiest part of this would be to ensure its innocuity to the rest of the infrastructure...

1.5 What would not be (directly) handled by the gaming platform

Matchmaking and ranking systems would be game dependent, and thus either completely left out of the gaming platform, or provided by additional specific Server parts of these games.

CHAPTER 2

Design

TODO

CHAPTER 3

Indices and Tables

- genindex
- modindex
- search