

## **Alpha Gomoku Second Iteration**

Team: LongNameWillBeRemembered

Chengqi Dai (cd3046), Yiqian Wang (yw3225), Wenbo Song (ws2505), Zhongkai Sun (zs2341)

Fall 2018 - COMS W4156 Advance Software Engineering - Prof. Gail Kaiser

In second iteration, we add implement for more users to play online gomoku.

For the online part, A player can check in a room for online game, and the room will be assigned a room ID. Other players can join a room if there is only one person in it by enter the specific room ID. Also, the players can choose random match option to play with another player, and a room will be automatic allocated for them. To make our service faster, we store the room information using Redis.

For the offline part, we add a setting manual to choose the size of chessboard and the font type. Also, users can pause or unpaue background music by press SPACE in keyboard.

The user stories we have complete:

User Story 1:

Title: user interface

Actor(s): game players

Description: As a user, I want to observe a pretty user interface, and I want to choose font styles and the size of chessboard.

Basic flow: Choose a specific size of chessboard

Alternate flows: If they don't make choices, it will come out a 15\*15 chessboard and a font type in "Times New Roman".

User Story 2:

Title: music

Actor(s): game players

Description: As a user, I want to have a delighted background music while I'm playing, so my conditions of satisfaction are determined by whether I'm into that song, and I can turn off the music and sound effects if I don't like it.

Basic flow: The music will play automatically, so the user can turn off the music if they don't like it.

Alternate flows: The music will continuously replay until the game end.

User Story 3:

Title: online game

Actor(s): game players

Description: As a user, I want to play game with my friends online, so I want to make sure we can check in a room to play game, and the pieces on the chessboard can correctly display.

Basic flow: Create a room with room number, and play game with another player

Alternate flows: If a room number has exist, the room cannot be created, or if the game has already started, other people cannot join the game, but they can watch the game instead.

Unit-test:

The equivalence partitions and boundary conditions necessary to unit-test:

In our unit-test, we have checked the invalid partitions that position is out of range, input value is out of range and two inputs have occupance conflict. For boundary conditions, we have three black wins (five in a row, five in a column and five in a polyline) and one white win (five in a row).

Test cases for loop initiation, continuation and termination:

Since we have a nested loop in our program, we test the inside loop first to test cases for loop initiation, continuation and termination. We set up a number of valid inputs first. Then, the inside loop will check if there is five row in a line by input a piece value, x & y position and direction from outer loop. We test the inside loop first for different inputs by our test case to test if it could be terminated and give a correct return value. Then we check the outer loop to test if the outer loop works correctly.

post-commit:

The detail for post-commit test is inside the file .travis.yml including language, version, install and requirements.

We use Coveralls to add a coverage tool to the post-commit CI process.

Link:

GitHub:

<https://github.com/OliviaWYQ/Gomoku-Desktop>

Travis-CI

<https://travis-ci.org/OliviaWYQ/Gomoku-Desktop>

Coveralls

<https://coveralls.io/github/OliviaWYQ/Gomoku-Desktop>