厦門大學

信息学院软件工程系

《JAVA 程序设计》实验报告

实验 15

姓名：黄勖

学号：22920212204392

学院：信息学院

专业：软件工程

完成时间：2023.6.6

一、实验目的及要求

- 熟悉多线程编程和网络编程

二、实验题目及实现过程

实验环境：Windows 10 21H2、jdk17、javafx scene builder、utf-8 编码

题目一

（一） 实验题目

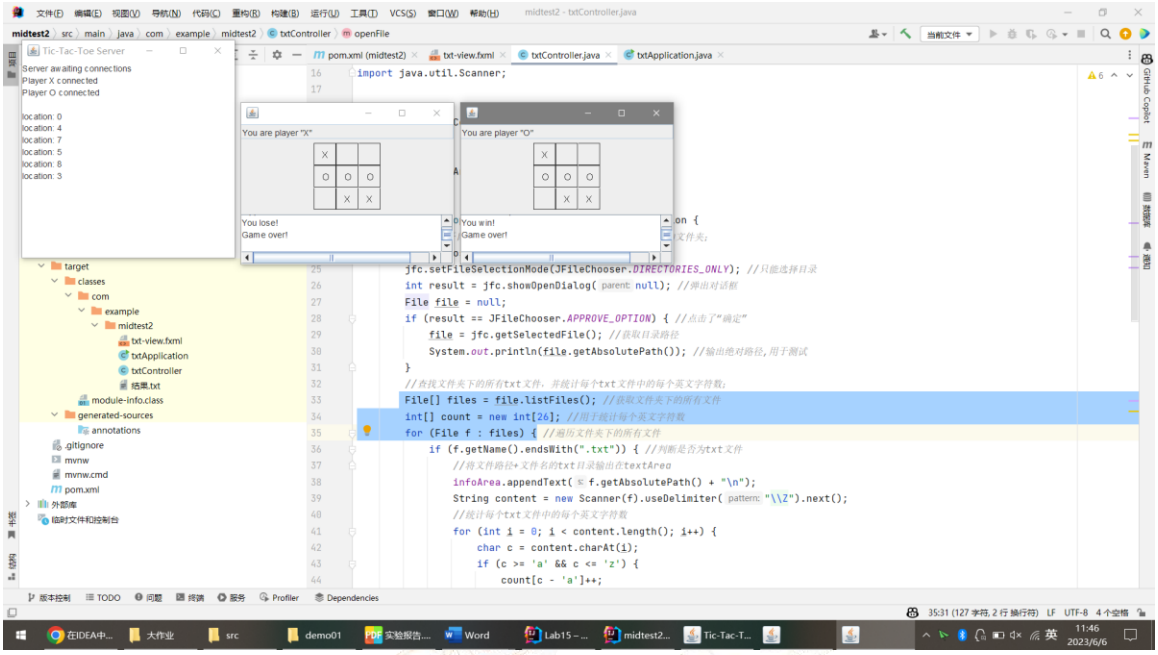
◆ 修改 TicTacToe 程序，补充“判断游戏结束”部分代码。。

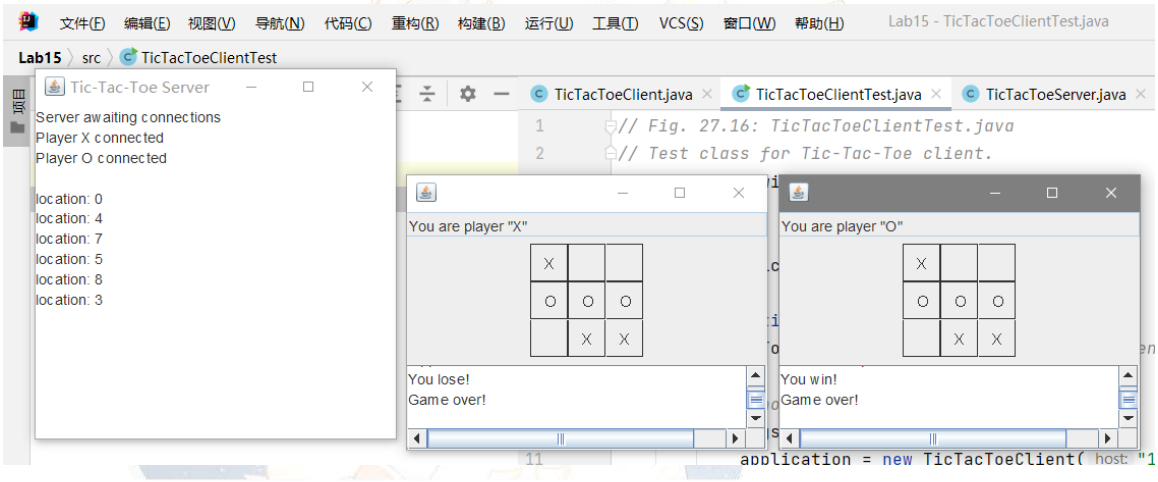
（二） 实现过程

思路：TicTacToeServer 实现服务器的功能，它创建 ServerSocket 对象，并初始化由两个玩家对象组成的玩家数组。Execute 方法中调用 accept 函数与客户端建立连接，创建玩家对象。isGameOver 函数根据 board 数组的情况判断游戏是否结束，三个相同的符号相连则游戏结束。Player 类实现 runnable 接口，是服务端的两个线程与两个客户端通信，其重写的 run 方法在输出流中传递连接、有效等信息。实现游戏结束要对 run 方法里的 while 函数进行改动，增加判断条件若游戏没有结束再判断这一步棋是否有效，没有结束则改变 board 的值并向输出流通信。若游戏结束则判断 loser 变量是否为 1，此变量用于区分赢家和输家（1 为输家，0 为赢家），若为赢家则设输家的 loser 变量为 1，向输家输入流输入失败信息后跳出循环；若为输家则直接跳出循环。

（三） 过程截图

最终结果（全屏截图）


Tic-Tac-Toe Server
Server awaiting connections
Player X connected
Player O connected
location: 0
location: 4
location: 7
location: 5
location: 8
location: 3
You are player "X"
You are player "O"
You lose!
Game over!
You win!
Game over!


Lab15 - TicTacToeClientTest.java
Tic-Tac-Toe Server
Server awaiting connections
Player X connected
Player O connected
location: 0
location: 4
location: 7
location: 5
location: 8
location: 3
You are player "X"
You are player "O"
You lose!
Game over!
You win!
Game over!

部分代码展示

TicTacToeClient.java × TicTacToeClientTest.java × TicTacToeServer.java × TicTacToeServerTest.java ×

```
155
156     // place code in this method to determine whether game over
        3 个用法
157     public boolean isGameOver() {
158         if (board[0].equals(board[1]) && board[1].equals(board[2])
159                 && (board[0].equals(MARKS[PLAYER_X]) || board[0].equals(MARKS[PLAYER_O])))
160             return true;
161         if (board[3].equals(board[4]) && board[4].equals(board[5])
162                 && (board[3].equals(MARKS[PLAYER_X]) || board[3].equals(MARKS[PLAYER_O])))
163             return true;
164         if (board[6].equals(board[7]) && board[7].equals(board[8])
165                 && (board[6].equals(MARKS[PLAYER_X]) || board[6].equals(MARKS[PLAYER_O])))
166             return true;
167         if (board[0].equals(board[3]) && board[3].equals(board[6])
168                 && (board[0].equals(MARKS[PLAYER_X]) || board[0].equals(MARKS[PLAYER_O])))
169             return true;
170         if (board[1].equals(board[4]) && board[4].equals(board[7])
171                 && (board[1].equals(MARKS[PLAYER_X]) || board[1].equals(MARKS[PLAYER_O])))
172             return true;
173         if (board[2].equals(board[5]) && board[5].equals(board[8])
174                 && (board[2].equals(MARKS[PLAYER_X]) || board[2].equals(MARKS[PLAYER_O])))
175             return true;
176         if (board[0].equals(board[4]) && board[4].equals(board[8])
177                 && (board[0].equals(MARKS[PLAYER_X]) || board[0].equals(MARKS[PLAYER_O])))
178             return true;
179         if (board[2].equals(board[4]) && board[4].equals(board[6])
180                 && (board[6].equals(MARKS[PLAYER_X]) || board[6].equals(MARKS[PLAYER_O])))
181             return true;
182         return false; // this is left as an exercise
183     } // end method isGameOver
```

三、实验总结与心得记录

通过本次实验，我对 Java 使用多线程、网络编程有了更深入的了解。Server 端创建了两个 player 对象相当于两个线程与不同的客户端进行通信，从而可以做到两个玩家进行通信，客户端与服务端进行通信。

注意需要使用“允许多个实例”，才能多开客户端：