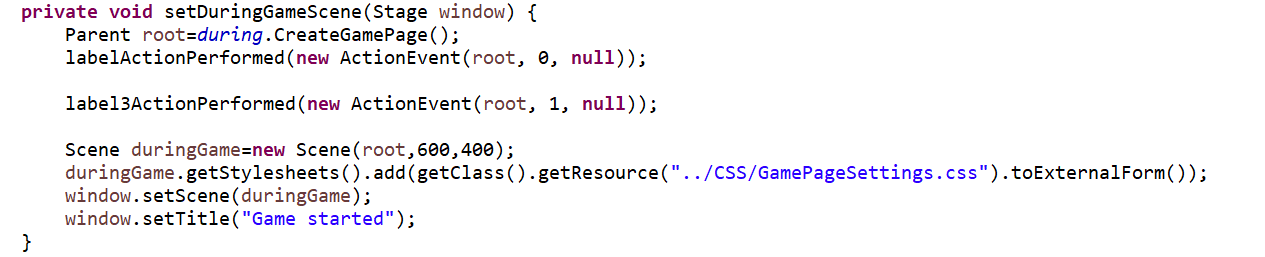
DRY principle

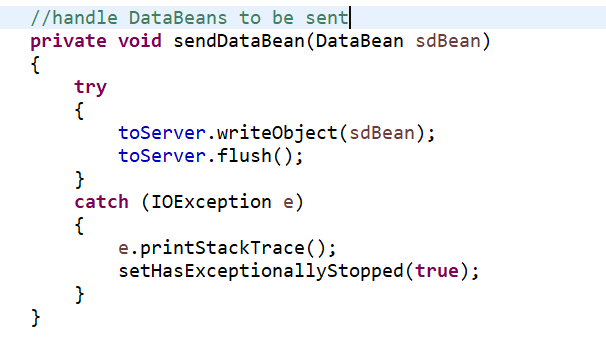
1. setDuringGameScene method(In ClientMain class)

Because the welcome page has 3 buttons, these buttons are set on the same action which is turn to the DuringTheGame page but with different mode number. We abstract these codes out into one method.



1. sendDataBean method(In Client class)

Because this game is an online game, client needs to send different kinds of data to the server. Notice that we have one abstract class called DataBean. And there are a lot of concrete classes extending it such as ChoiceBean, StartBean and so on. So, we simply abstract the duplicate codes into one method. When the client needs to send data to the server, we can simply call the sendDataBean method.



By following this principle, it is easy for us to reuse and maintain.

C&C principle

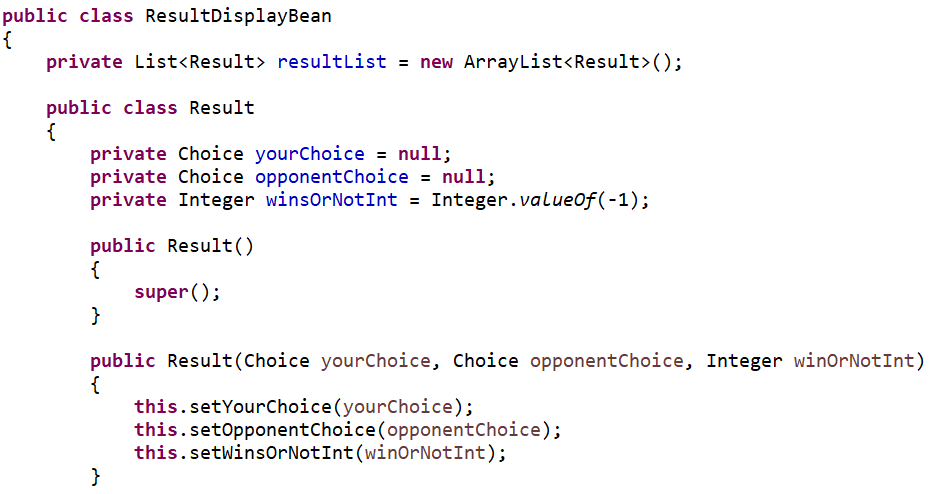
1. Complete

Each class should have exactly one role but similar type of behaviors can also add to that class to make it more complete.

There are always related behaviors that they will exist together in nature.

In our game, for client we have initialize( ) and terminate( ) method. For server, we have clientRegister( ) and clientDeregister(roomNo: int, uuid: UUID).

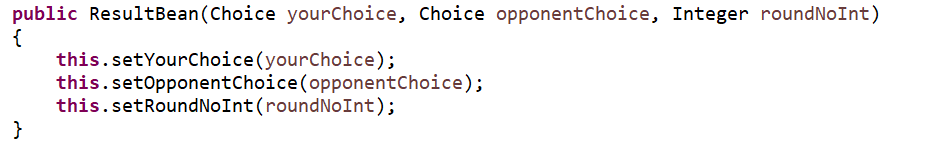
1. Consistent
2. The naming convention of the methods should be in the same manner.

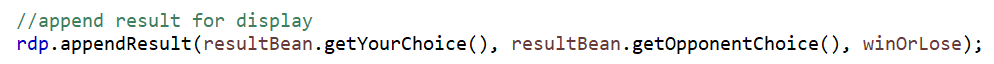
For the client side, we have the ResultDisplayBean class. For the server side, we have the ResultBean class. It’s not hard to see they are all in the same manner.

1. The parameters they(the methods) take should be in the same order.

Below are the examples I list. We can discover that the first parameter is always your choice and the second parameter is always the opponent’s choice.

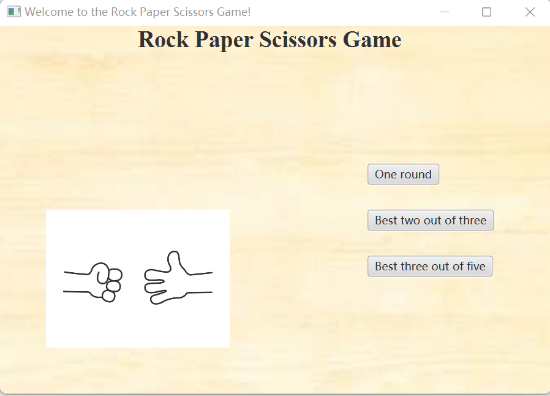


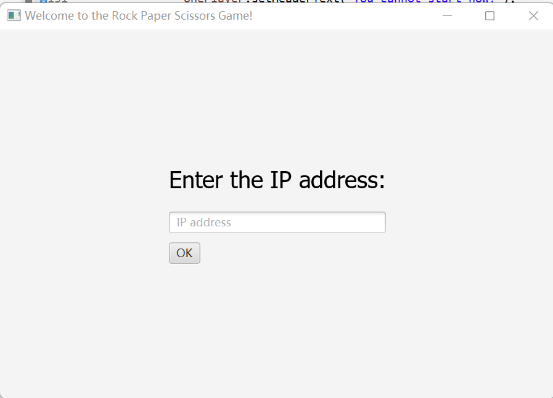




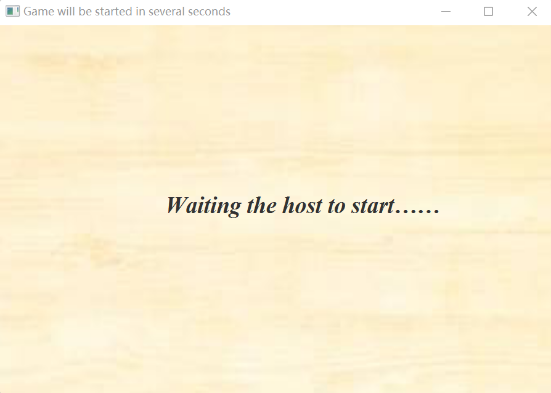
（3）The naming for methods and variables should be meaningful

Above pictures shows us the sendResultBean method and ResultBean constructor. Their names are quite meaningful for us to get the information directly about what these methods do for us.

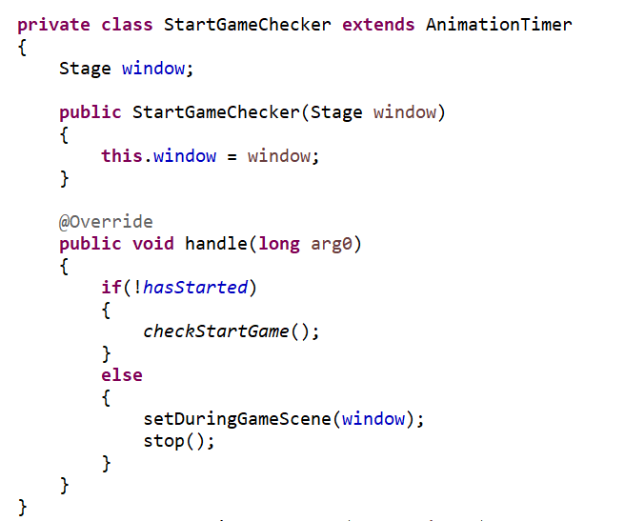
Controller(View)

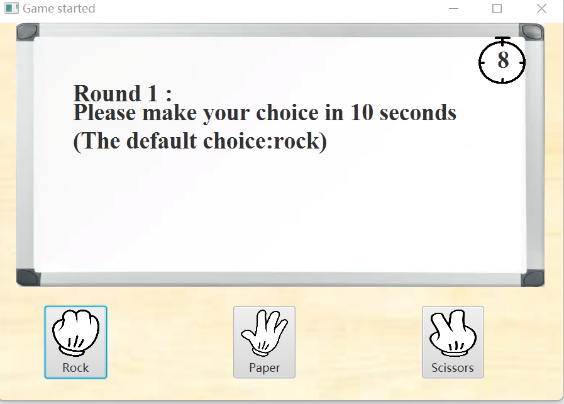
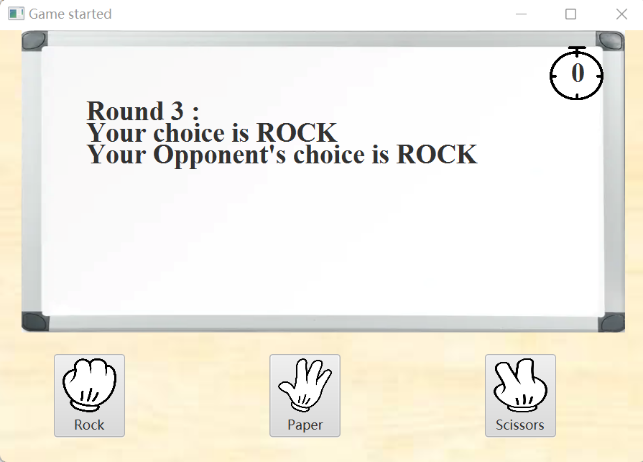
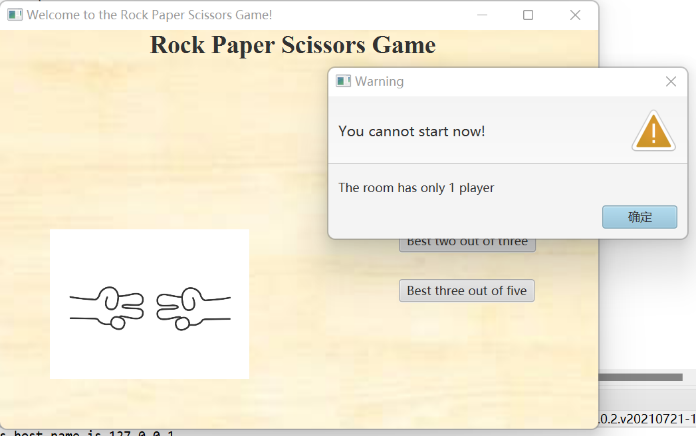


Client.isHost( )



1. At the beginning, 2 players need to input the IP addresses. After they input the IP address, the player who enter the game firstly will become the host and he will have the unique ability to choose the mode number. The second player will simply turn to the waiting page. The OK button is set on action to turn to the different page based on the if condition statement.
2. After the host choose the mode, the non-host player will automatically turn to the DuringTheGame page. This function is achieved by setting an animation timer. It continuously checks whether the host starts game. When the host starts game, the non-host player will automatically turn to the DuringTheGame page at the same time. And also, for ending the game, it use the same method.



1. Let’s concentrate on the DuringTheGame page. Please look at top right-hand corner of this page, you can see there is a count-down timer. This count-down timer counts from 10 to 0. Actually, this timer is consisted of one label and one clock imageview. The label is put on the top of the imageview. This function is achieved by defining a method which contains a timer. This timer is assigned a timertask and this timertask will be executed per second. And also for the reminder, we use the same method. But here period(*time between successive task executions*) is 11 seconds. There is also one situation use this method too. Usually the player needs to make choice in 10 seconds. If the player forgets to choose one of the gestures, we will help them choose the rock by default. Notice here the delay(*delay before task is to be executed*) is 10 seconds and the period is 11 seconds. If the player has made choice in the 10 seconds, we will simply set the makeChoice boolean value to be false. Otherwise, we will help them choose rock in addtion.
2. Notice that the result will also be displayed on the white board. When the timer has the number 0, the result will be displayed on the whiteboard. This is achieved by setting up an animation timer. And this animation timer will start when the count-down timer’s number is 0.
3. At the last, we list 2 alert to talk about. The first one is one player condition. Because this online game needs 2 players. If there is only one player, he can’t start the game and he will receive the warning message. This is simply set an if condition statement in the button setOnAction method. The second one is about the opponent exits the game. If the opponent exits the game, another player will receive the warning message. No matter the player chooses close or ok for this alert, this game will be ended. This is achieved by setting an animation timer. This animation timer continuously checks whether the client exits the game. If the client exits the game, another player will receive the warning message at the same time.

