CISC 650 – Computer Network, Project: Network Jeopardy Protocol

Author: Long Chen

Basic Idea:

In this program, the server needs to connect at least two clients to send messages concurrently and the server and clients can communicate with each other. This means we need to handle multiple connections concurrently.

The last project is the application of the basic socket programming mode. In the basic socket programming model (i.e. basic server and client model), server handles only one client at a time, which is a big assumption if we just want to develop a scalable server model. In order to achieve multiple connections concurrently, this program uses select() linux command. select() model allows to check for I/O completion on more than one socket. In Python, we can use the selectors module in the standard library. Even though, by using select(), we’re not able to run concurrently, select() allows to monitor multiple file descriptors, waiting until one of the file descriptors become active. For example, if there is some data to be read on one of the sockets select will provide that information. Select works like an interrupt handler, which gets activated as soon as any file descriptor sends any data.

The following is an example of using finite-state-machine to display TCP connection (both the Game-master and player are TCP finite state machine).

手机屏幕截图

描述已自动生成

Here’s a simple description of the game procedure steps.

Server Part:

1. Game-master starts server. Server stays idle and wait for clients’ subscription message. When receives CM\_SUBSCRIBE message and in COLLECT\_SUBSCRIBTIONS state. The server is “activated” and changes state to GAME\_IN\_PROGRESS state and writes the SM\_NEW\_GAME message to the client.

2. When server receives CM\_NEW\_ROUND or CM\_ANSWER\_SHOW message and in GAME\_IN\_PROGRESS, ANSWER\_SHOW state, the server is “activated” and changes state to ROUND\_IN\_PROGRESS state and writes the SM\_NEW\_ROUND message to the client.

3. When server receives CM\_CATEGORY message and in ROUND\_IN\_PROGRESS state, the server is “activated” and changes state to WAIT\_RING state and writes the SM\_CATEGORY message to the client.

4. When server receives CM\_CATEGORY\_RECV message and in WAIT\_RING state, the server is “activated” and changes state to WAIT\_FOR\_RING state and writes the SM\_QUESTION message to the client.

5. When server receives CM\_RING message and in WAIT\_FOR\_RING state, the server is “activated” and changes state to WAIT\_FOR\_CM\_ANSWER state and writes the SM\_RING\_CLIENT message to the client

6. When server receives CM\_ANWER message and in WAIT\_FOR\_CM\_ANSWER state, the server is “activated” and changes state to ANSWER\_SHOW state and writes the SM\_ANSWER message to the client.

Client Part:

1. Player starts client. When client receives SM\_WELCOME message and in JOINING\_GAME state, the client is “activated” and changes state to GAME\_IN\_PROGRESS and writes the CM\_SUBSCRIBE message to the server.

2. When client receives SM\_NEW\_GAME message and in GAME\_IN\_PROGRESS state, the client is “activated” and change state to ROUND\_IN\_PROGRESS and writes the CM\_NEW\_ROUND message to the server.

3. When client receives SM\_NEW\_ROUND message and in ROUND\_IN\_PROGRESS or ANSWER\_SHOW state, the client is “activated” and change state to CATEGORY\_SELECTED and writes the CM\_CATEGORY message to the server.

4. When client receives SM\_CATEGORY message and in CATEGORY\_SELECTED state, the client is “activated” and change state to DISPLAY\_QUESTION and writes the CM\_CATEGORY\_RECV message to the server.

5. When client receives SM\_QUESTION message and in DISPLAY\_QUESTION state, the client is “activated” and change state to DISPLAY\_QUESTION and writes the CM\_RING message to the server.

6. When client receives SM\_RING\_CLIENT message and in DISPLAY\_QUESTION state, the client is “activated” and change state to ANSWER\_SHOW and writes the CM\_ANSWER message to the server.

7. When client receives SM\_ANSWER message and in ANSWER\_SHOW state, the client is “activated” and change state to ANSWER\_SHOW and writes the CM\_ANSWER\_SHOW message to the server.

Here’s a manual of how to run my code.

In order to run code, first open three terminals (here I use MAC system) and in first terminal, use command

```

python3 server.py

```

you will see a “starting up on 127.0.0.1 port 33333” message and then just leave this terminal there. Then in the second terminal, use command

```

python3 client.py

```

follow the prompt and enter the first player’s name and leave this terminal there. You will find the server terminal will receive and show message as bellow:

图片包含 游戏机

描述已自动生成

This means the first player is successfully connected and the server is waiting for the second player. Now in the third terminal, use command

```

python3 client.py

```

follow the prompt and enter the second player’s name and leave this terminal there. You will find after you enter the second player’s name, the server received the message that two players are ready and sent back a large paragraph of players’ information and the game rules.

Two players can receive the server’s information concurrently, like the screenshot below:

player 1:

手机屏幕截图

描述已自动生成

Player 2:

手机屏幕截图

描述已自动生成

At this time, the server randomly picks one player and asks this player (here player 2) to select a category. Then go back to our terminal 3 (Player 2), and follow the direction and enter the category number, 0 or 1.

After entering the number 0 or 1, another message received from the server, and the same, both of two players can receive this message. These are some screenshots of two players:

Player 1:

手机屏幕截图

描述已自动生成

Player 2:

手机屏幕截图

描述已自动生成

At this time, it is a ring-the-bell part, **and at this part, both players need to ring the bell** (just enter ID number: 1 or 2), but only the first one who rings the bell can answer the question. If player 2 rings bell, for example, both players will receive messages but not the same. Here are some screenshots:

Player 2 (ring the bell first)

手机屏幕截图

描述已自动生成

Player 1:

手机屏幕截图

描述已自动生成

At this round, only player 2 can enter the answer while player 1needs to wait. After player 2 enter the answer, the server will send back the correct answer and both player 1 and player 2 can receive this message. After received the answer, both player 1 and player 2 will jump to a new round and repeat the category selected and answer question part. Here are some screenshots:

Player 2:

手机屏幕截图

描述已自动生成

Player 1:

手机屏幕截图

描述已自动生成

The last point I want to mention here is at the server’s terminal (the first terminal that we open), each time the server sends a message, both of the players need to reply one back. In the server’s terminal, it shows each state’s number and received message from the player. Here’s a screenshot that we can see clearly.

手机屏幕截图

描述已自动生成

That’s all about this game and enjoy!