Intro:

In this tutorial we are going to learn how to turn our single player game into a multiplayer game.

Clip 1:

First so that we could differentiate between the game clients we have to get the player’s unique Id.  
We do this by going to the tickTacToeMain class (pause)

Then storing T.custom of CUSTOM\_INFO\_KEY\_myUserId in our myUserId parameter.

Clip 2:

After sending us the gotCustomInfo callback, the server will wait for the rival player to connect, and then he will send us a gotMatchStarted callback, if you remember that in the previous tutorial we initialized our game from the constructor, we now initialize it from the gotMatchStarted callback. This is made so that the players could play a couple of games consecutively.

Besides letting you know the game has started ,the gotMatchStarted callback transfers some key elements, allPlayerIds which is an array containing all the players in the game, finishedPlayerIds is an array of all the players that left the game and serverEntries which will be discussed in chapter 7.

Clip 3:

Remember in the first tutorial that we took each game move from the game logic, and instead of directly committing it, we sent it to the TickTacToeMain class and only then committed it. Now we are going to send it to the server instead of committing it at all, we do it by calling the doStoreState function, the doStoreState function has to get an Array of User Entry elements, so we take our game Move and place it in a UserEntry by calling the static create function of UserEntry, with a key and value. You can later use these keys to delete entries from the state.

Clip 4:

After one of the players calls doStoreState, all users get a gotStateChanged callback containing the entries the player sent, we check the key and value are as expected, and perform the move.

From this point on, any change to the game’s logic will only occur through changes sent to the server, and not in a direct way

Clip 5:

Now let’s go to the emulator and see what our changes have made (pause)

Notice that now any move happening in one player window is reflected in the other.