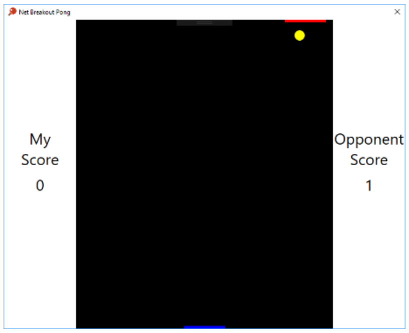
Network Pong Game

Skills Used

- C# and XAML
- TCP
- JSON
- Multi-threading



Description

A basic pong game that can be played over the network. This was some of my first exposure to C# and XAML.

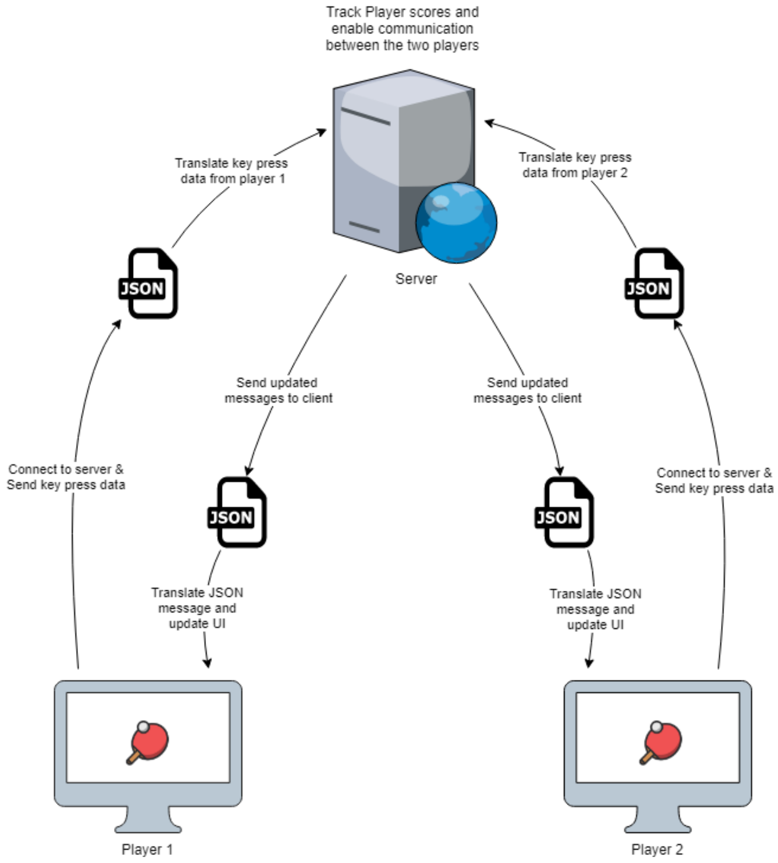
Implementation

We used a Client/Server architecture with a server that accepts connection requests from two clients to play a game.

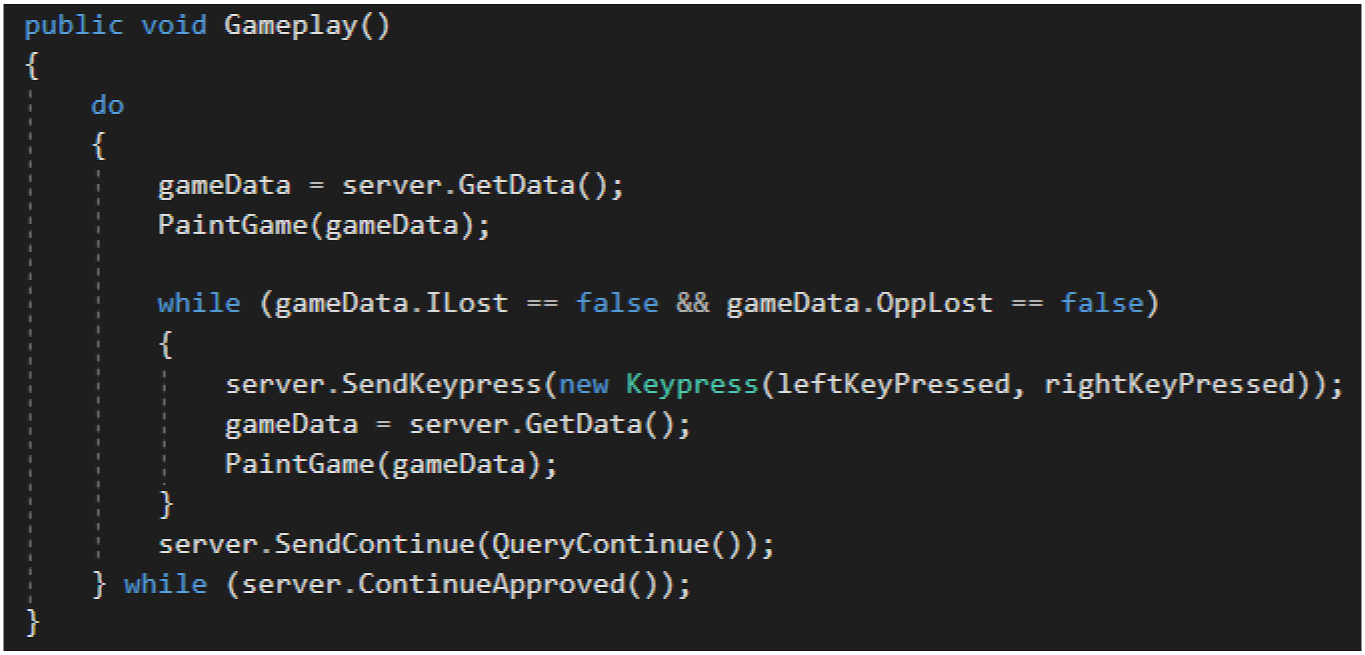
The server receives the key presses of the two users and then sends updated locations to each client in the Json format.

Each client will then deserialize the Json message and update its UI to match the updated information.

Diagrams


Track Player scores and enable communication between the two players
Server
Translate key press data from player 1
Translate key press data from player 2
JSON
JSON
Send updated messages to client
Send updated messages to client
Connect to server & Send key press data
Connect to server & Send key press data
JSON
JSON
Translate JSON message and update UI
Translate JSON message and update UI
Player 1
Player 2

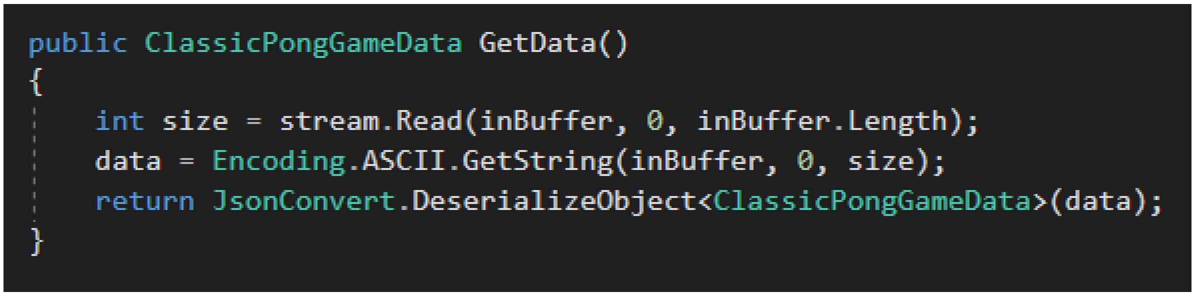
CODE SNIPPETS



```
public void Gameplay()
{
    do
    {
        gameData = server.GetData();
        PaintGame(gameData);

        while (gameData.ILost == false && gameData.OppLost == false)
        {
            server.SendKeypress(new Keypress(leftKeyPressed, rightKeyPressed));
            gameData = server.GetData();
            PaintGame(gameData);
        }
        server.SendContinue(QueryContinue());
    } while (server.ContinueApproved());
}
```

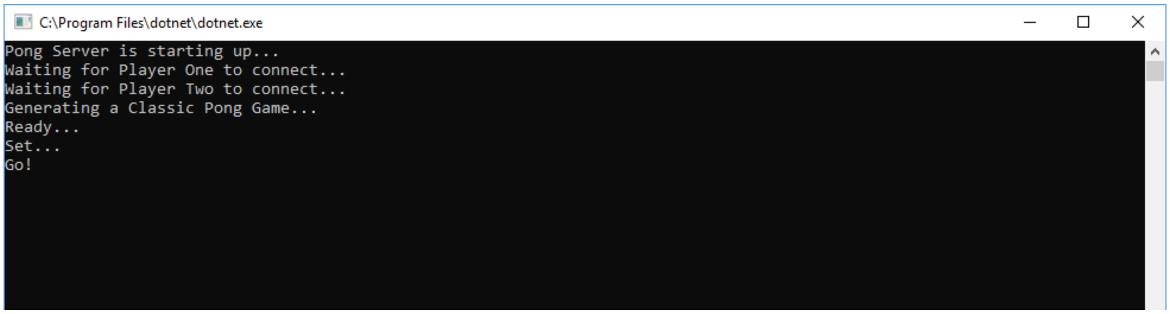
This is the method that drives the game. The client is continually pulling data from the server and updating its UI with the new locations of the ball and paddles. We continue gameplay until one of the players loses and ask each player if they would like to continue playing.



```
public ClassicPongGameData GetData()
{
    int size = stream.Read(inBuffer, 0, inBuffer.Length);
    data = Encoding.ASCII.GetString(inBuffer, 0, size);
    return JsonConvert.DeserializeObject<ClassicPongGameData>(data);
}
```

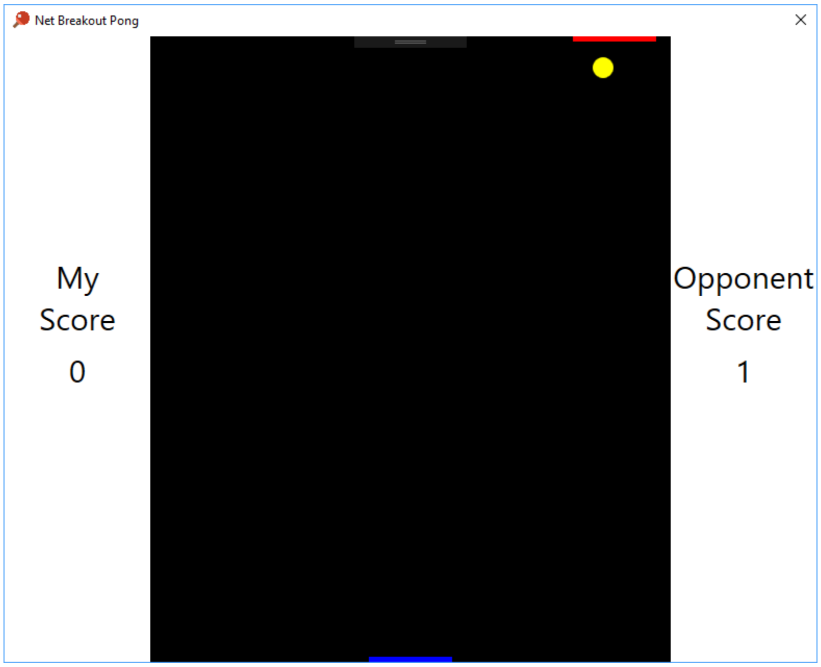
This is an example of how we passed data from server to client. The GetData method asks the server for the updated locations and the server sends a json message to the client containing all the gameplay data. The client can then deserialize this data and update its UI accordingly.

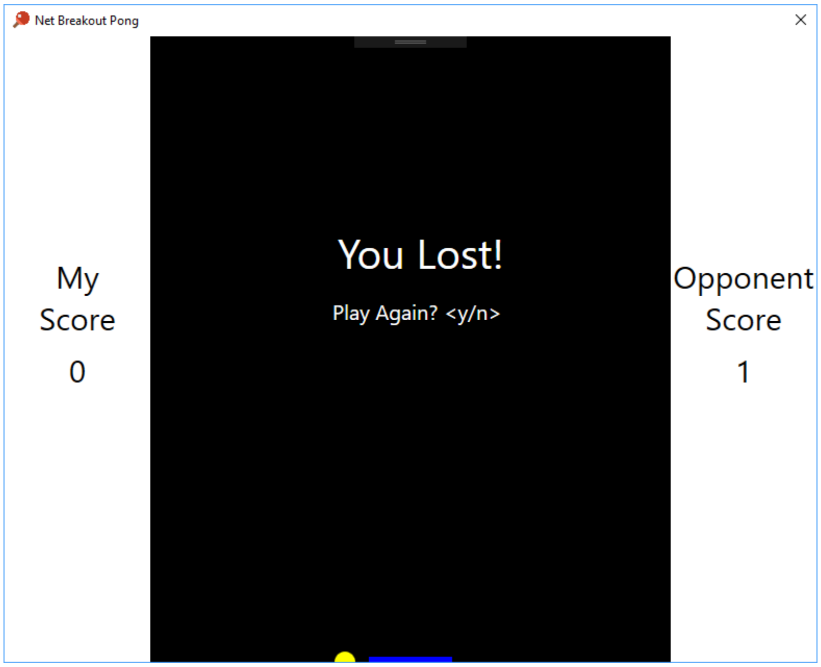
SCREENSHOTS



```
Pong Server is starting up...
Waiting for Player One to connect...
Waiting for Player Two to connect...
Generating a Classic Pong Game...
Ready...
Set...
Go!
```

Clients


Net Breakout Pong
My Score
0
Opponent Score
1


Net Breakout Pong
My Score
0
You Lost!
Play Again? <y/n>
Opponent Score
1