Unity- Chat system for players- C# script

1. We must create a UI for the chat server to run on. My best advice is to run this application inside a panel that is in the scene, therefore the client chat can be run inside of a game you are playing. Create a panel by right clicking, going to UI and Click “Create Panel”. Create a panel inside the panel. Create a user input field inside this panel (i.e: hierarchy right click, “ UI”, “input field”. Anchor to left and copy to right side. Create a button underneath these fields. Name it connect. Name the left input “HostInput” (the input field in the hierarchy) and name the right one “PortInput”. (127.0.0.1 Is localhost)

6321 for port

1. Make a folder inside scripts. One called server and one call client. In “server” create a script called “Server.cs”. Make sure clientName under ServerClient can be interchanged, you can even set it to a username name in sql databases, which are included in another document series.

using UnityEngine;

using System.Collections;

using System.Net.Sockets;

using System.Collections.Generic;

using System;

using System.Net;

public class Server : MonoBehaviour

{

private List<ServerClient> clients;

private List<ServerClient> disconnectList;

public int port = 6321;

private tcpListener server;

private bool serverStarted;

private void Start()

{

clients = new List<ServerClient>();

disconnectList = new List<ServerClient>();

try

{

server = new TcpListener(IPAddress.any, port);

server.Start();

StartListening();

serverStarted = true;

Debug.Log(“Server has been started on port “ + port.ToString());

}

catch(Exception e)

{

Debug.Log(“Socket error: “ + e.Message);

}

private void Update()

{

if (!serverStarted)

return;

foreach (ServerClient c in clients)

{

// is the client still connected?

if(!IsConnected(c.tcp))

{

c.tcp.Close();

disconnectList.Add©;

continue;

}

// check for message from client

else

{

NetworkStream s = c.tcp.GetStream();

if(s.DataAvailable)

{

StreamReader reader = new StreamReader(s, true);

string data = reader.ReadLine();

if(data != null)

OnIncomingData(c, data);

}

}

}

}

private void StartListening()

{

server.BeginAcceptTcpClient(AcceptTcpClient, server);

}

private bool IsConnected(TcpClient c)

{

try

{

if (c != null && c.Client != null && c.Client.Connected)

{

if (c.Client.Poll(0, SelectMode.SelectRead))

{

return !(c.Client.Recieve(new byte[1], SocketFlags.Peek) == 0);

}

return true;

}

else

return false;

}

catch

{

return false;

}

}

private void AcceptTcpClient(IAsyncResult ar)

{

TcpListener listener = (TcpListener)ar.AsyncState;

clients.Add(new ServerClient(listener.EndAcceptTcpClient(ar)));

StartListening();

// send a message to everyone, say someone has connected

}

private void OnIncomingData(ServerClient c, string data)

{

Debug.Log(c.clientName + “ has sent the following message! : “ + data);

}

}

public class ServerClient

{

public TcpClient tcp;

public string clientname;

public ServerClient(TcpClient clientSocket)

{

clientName = “Guest”;

tcp = clientSocket;

}

1. Create an empty game object. Call it “Server”. Attach the script to this file. You can change the port number, and find the information about where the port is. This is the fundamentals. Read Part two.