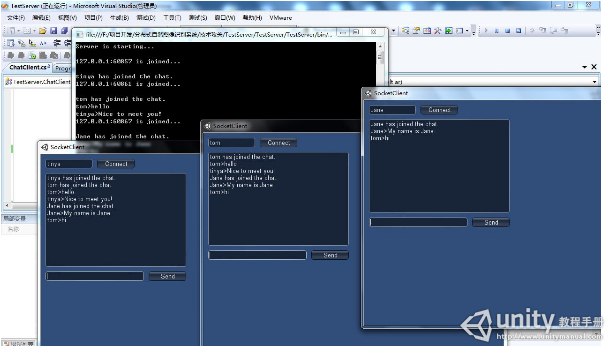
**Unity3D教程：实现基于Socket通讯的公共聊天室**

Posted on 2013年01月17日 by U3d / [Unity3D 基础教程](http://www.unitymanual.com/category/manual/unity3d-%e5%9f%ba%e7%a1%80%e6%95%99%e7%a8%8b)/被围观 544 次

多个客户端一同使用就是一个简单的公共聊天室。服务端为一个控制台程序使用C#实现，当然，在Unity3D中也相应地使用了C#语言实现客户端，服务端和客户端能实现消息的互通，当服务端接收到某客户端发送过来的消息时将会对客户端列表成员进行广播，这是公共聊天室的最基本的形式。Socket通讯是网络游戏最为基础的知识，因此这个实例能向有志投身于网游行业的初学者提供指导意义。



Unity3D教程：实现基于Socket通讯的公共聊天室

using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using System.Net.Sockets;  
namespace TestServer  
{  
class Program  
{  
// 设置连接端口  
const int portNo = 500;  
static void Main(string[] args)  
{  
// 初始化服务器IP  
System.Net.IPAddress localAdd = System.Net.IPAddress.Parse("127.0.0.1");  
// 创建TCP侦听器  
TcpListener listener = new TcpListener(localAdd, portNo);  
listener.Start();  
// 显示服务器启动信息  
Console.WriteLine("Server is starting...\n");  
// 循环接受客户端的连接请求  
while (true)  
{  
ChatClient user = new ChatClient(listener.AcceptTcpClient());  
// 显示连接客户端的IP与端口  
Console.WriteLine(user.\_clientIP + " is joined...\n");  
}  
}  
}  
}

**ChatClient.cs**

using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using System.Collections;  
using System.Net.Sockets;  
namespace TestServer  
{  
class ChatClient  
{  
public static Hashtable ALLClients = new Hashtable(); // 客户列表  
private TcpClient \_client; // 客户端实体  
public string \_clientIP; // 客户端IP  
private string \_clientNick; // 客户端昵称  
private byte[] data; // 消息数据  
private bool ReceiveNick = true;  
public ChatClient(TcpClient client)  
{  
this.\_client = client;  
this.\_clientIP = client.Client.RemoteEndPoint.ToString();  
// 把当前客户端实例添加到客户列表当中  
ALLClients.Add(this.\_clientIP, this);  
data = new byte[this.\_client.ReceiveBufferSize];  
// 从服务端获取消息  
client.GetStream().BeginRead(data, 0, System.Convert.ToInt32(this.\_client.ReceiveBufferSize), ReceiveMessage, null);  
}  
// 从客戶端获取消息  
public void ReceiveMessage(IAsyncResult ar)  
{  
int bytesRead;  
try  
{  
lock (this.\_client.GetStream())  
{  
bytesRead = this.\_client.GetStream().EndRead(ar);  
}  
if (bytesRead < 1)  
{  
ALLClients.Remove(this.\_clientIP);  
Broadcast(this.\_clientNick + " has left the chat");  
return;  
}  
else  
{  
string messageReceived = System.Text.Encoding.ASCII.GetString(data, 0, bytesRead);  
if (ReceiveNick)  
{  
this.\_clientNick = messageReceived;  
Broadcast(this.\_clientNick + " has joined the chat.");  
//this.sendMessage("hello");  
ReceiveNick = false;  
}  
else  
{  
Broadcast(this.\_clientNick + ">" + messageReceived);  
}  
}  
lock (this.\_client.GetStream())  
{  
this.\_client.GetStream().BeginRead(data, 0, System.Convert.ToInt32(this.\_client.ReceiveBufferSize), ReceiveMessage, null);  
}  
}  
catch (Exception ex)  
{  
ALLClients.Remove(this.\_clientIP);  
Broadcast(this.\_clientNick + " has left the chat.");  
}  
}  
// 向客戶端发送消息  
public void sendMessage(string message)  
{  
try  
{  
System.Net.Sockets.NetworkStream ns;  
lock (this.\_client.GetStream())  
{  
ns = this.\_client.GetStream();  
}  
// 对信息进行编码  
byte[] bytesToSend = System.Text.Encoding.ASCII.GetBytes(message);  
ns.Write(bytesToSend, 0, bytesToSend.Length);  
ns.Flush();  
}  
catch (Exception ex)  
{

}  
}  
// 向客户端广播消息  
public void Broadcast(string message)  
{  
Console.WriteLine(message);  
foreach (DictionaryEntry c in ALLClients)  
{  
((ChatClient)(c.Value)).sendMessage(message + Environment.NewLine);  
}  
}  
}  
}

**Unity3d客户端的代码ClientHandler.cs：**

using UnityEngine;  
using System.Collections;  
using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Text;  
using System.Net.Sockets;  
public class ClientHandler : MonoBehaviour  
{  
const int portNo = 500;  
private TcpClient \_client;  
byte[] data;

// Use this for initialization  
void Start ()  
{

}

// Update is called once per frame  
void Update ()  
{

}

public string nickName = "";  
public string message = "";  
public string sendMsg = "";

void OnGUI()  
{  
nickName = GUI.TextField(new Rect(10, 10, 100, 20), nickName);  
message = GUI.TextArea(new Rect(10, 40, 300, 200), message);  
sendMsg = GUI.TextField(new Rect(10, 250, 210, 20), sendMsg);

if(GUI.Button(new Rect(120, 10, 80, 20), "Connect"))  
{  
//Debug.Log("hello");

this.\_client = new TcpClient();  
this.\_client.Connect("127.0.0.1", portNo);  
data = new byte[this.\_client.ReceiveBufferSize];  
//SendMessage(txtNick.Text);  
SendMessage(nickName);  
this.\_client.GetStream().BeginRead(data, 0, System.Convert.ToInt32(this.\_client.ReceiveBufferSize), ReceiveMessage, null);  
};

if(GUI.Button(new Rect(230, 250, 80, 20), "Send"))  
{  
SendMessage(sendMsg);  
sendMsg = "";  
};  
}

public void SendMessage(string message)  
{  
try  
{  
NetworkStream ns = this.\_client.GetStream();  
byte[] data = System.Text.Encoding.ASCII.GetBytes(message);  
ns.Write(data, 0, data.Length);  
ns.Flush();  
}  
catch (Exception ex)  
{  
//MessageBox.Show(ex.ToString());  
}  
}

public void ReceiveMessage(IAsyncResult ar)  
{  
try  
{  
int bytesRead;  
bytesRead = this.\_client.GetStream().EndRead(ar);  
if (bytesRead < 1)  
{  
return;  
}  
else  
{

Debug.Log(System.Text.Encoding.ASCII.GetString(data, 0, bytesRead));

message += System.Text.Encoding.ASCII.GetString(data, 0, bytesRead);  
}  
this.\_client.GetStream().BeginRead(data, 0, System.Convert.ToInt32(this.\_client.ReceiveBufferSize), ReceiveMessage, null);

}  
catch (Exception ex)  
{

}  
}  
}