**socket编程 tcp/ip**

Posted on 2013年07月26日 by U3d / [Unity3D脚本/插件](http://www.unitymanual.com/category/script) /被围观 12 次

客户端代码

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| 01 | **using** System; |
| 02 | **using** System.Collections.Generic; |
| 03 | **using** System.Text; |
| 04 | **using** System.Net; |
| 05 | **using** System.IO; |
| 06 | **using** System.Net.Sockets; |
| 07 | **namespace** client |
| 08 | { |
| 09 | **class** Program |
| 10 | { |
| 11 | **static** **void** Main(**string**[] args) |
| 12 | { |
| 13 | **try** |
| 14 | { |
| 15 | *//创建客户端socket，使用tcp通信协议* |
| 16 | TcpClient TcpClient = new TcpClient(); |
| 17 | TcpClient.Connect("127.0.0.1", 8888); *//指定地址与端口号* |
| 18 |  |
| 19 | Console.WriteLine("Connect Complete"); |
| 20 | Console.WriteLine("Input string:"); |
| 21 |  |
| 22 | *//Connect方法使用指定的IP地址和端口号将客户端连接到远程TCP主机* |
| 23 | *//读入控制台输入的文字* |
| 24 | **String** str = Console.ReadLine(); |
| 25 |  |
| 26 | Stream stm = TcpClient.GetStream(); |
| 27 |  |
| 28 | *//Unity3D教程手册：www.unitymanual.com* |
| 29 | *//定义数据流，用于发送和接收数据* |
| 30 | ASCIIEncoding **AS** = new ASCIIEncoding(); |
| 31 |  |
| 32 | *//将字符串转换为byte类型* |
| 33 | **byte**[] b = **AS**.GetBytes(str); |
| 34 |  |
| 35 | *//Write(参数1，参数2，参数3)表示向服务端发送字符串* |
| 36 | *//参数1指将此数组复制到当前流，参数2指从零开始的字节偏移量，参数3指要写入当前流的字节数(即字符串长度)* |
| 37 | stm.Write(b, 0, b.Length); |
| 38 |  |
| 39 |  |
| 40 | Console.WriteLine("Send Complete"); |
| 41 | *//Unity3D教程手册：www.unitymanual.com* |
| 42 |  |
| 43 | *//stm.Read在当前流中读入服务端发来的响应信息，其参数与Write方法参数一致，k值为读入字符串的长度* |
| 44 | **byte**[] bb = new **byte**[100]; |
| 45 | **int** k = stm.Read(bb, 0, 100); |
| 46 | **for** (**int** i = 0; i < k; i++) |
| 47 | Console.Write(Convert.ToChar(bb[i])); |
| 48 | TcpClient.Close(); |
| 49 |  |
| 50 |  |
| 51 |  |
| 52 | *//释放TcpClient实例，并不关闭基础连接* |
| 53 | Console.ReadLine(); |
| 54 | } |
| 55 | **catch** (Exception e) |
| 56 | { |
| 57 | Console.WriteLine("Error!" + e.StackTrace); |
| 58 | } |
| 59 | } |
| 60 | } |
| 61 | } |

服务端代码

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| 01 | **using** System; |
| 02 | **using** System.Collections.Generic; |
| 03 | **using** System.Text; |
| 04 | **using** System.Net; |
| 05 | **using** System.Net.Sockets; |
| 06 | **namespace** server |
| 07 | { |
| 08 | **class** Program |
| 09 | { |
| 10 | **static** **void** Main(**string**[] args) |
| 11 | { |
| 12 | **try** |
| 13 | { |
| 14 | *//创建listener，ip-地址，port-端口* |
| 15 | IPAddress Ip = IPAddress.Parse("127.0.0.1"); |
| 16 | TcpListener TcpList = new TcpListener(Ip, 8888); |
| 17 | TcpList.Start(); |
| 18 |  |
| 19 |  |
| 20 | Console.WriteLine("Server start!"); |
| 21 | Console.WriteLine("Ip address:" + TcpList.LocalEndpoint); |
| 22 | *//LocalEndpoint获取服务端(即本地)地址与端口等信息* |
| 23 | Console.WriteLine("Wait"); |
| 24 |  |
| 25 | *//获取监听到的socket* |
| 26 | Socket Soc = TcpList.AcceptSocket(); |
| 27 |  |
| 28 | Console.WriteLine("Received Connection:" + Soc.RemoteEndPoint); |
| 29 |  |
| 30 |  |
| 31 | *//接收到的信息 //Unity3D教程手册：www.unitymanual.com* |
| 32 | **byte**[] b = new **byte**[100]; |
| 33 | *//Soc.Receive(b)从socket接收数据，将数据存入接收缓冲区列表中，k的值为该数据的长度* |
| 34 | **int** k = Soc.Receive(b); |
| 35 | Console.WriteLine("Received data from client:"); |
| 36 | **for** (**int** i = 0; i < k; i++) |
| 37 | *//Convert.ToChar(b[i])将数组b转换基本数据类型为char的类型并输出* |
| 38 | Console.Write(Convert.ToChar(b[i])); |
| 39 |  |
| 40 | *//发送信息* |
| 41 | *//ASCIIEncoding表示Unicode字符的ASCII字符编码类* |
| 42 | ASCIIEncoding **AS** = new ASCIIEncoding(); |
| 43 |  |
| 44 | *//Unity3D教程手册：www.unitymanual.com* |
| 45 | *//Soc.Send向客户端发送数据，AS.GetBytes()获得括号中字符串的bytes值* |
| 46 | Soc.Send(**AS**.GetBytes("Received data!")); |
| 47 |  |
| 48 |  |
| 49 |  |
| 50 |  |
| 51 | *//关闭连接并释放所有关联的资源* |
| 52 | Soc.Close(); |
| 53 |  |
| 54 |  |
| 55 |  |
| 56 |  |
| 57 | *//关闭侦听* |
| 58 | TcpList.Stop(); |
| 59 |  |
| 60 |  |
| 61 |  |
| 62 |  |
| 63 | *//等待输入，起到暂停的作用* |
| 64 | Console.ReadLine(); |
| 65 |  |
| 66 | } |
| 67 | **catch**(Exception e) |
| 68 | { |
| 69 | Console.WriteLine("Error!" + e.StackTrace); |
| 70 | *//获取当前异常发生时调用堆栈上的帧的字符串* |
| 71 | } |
| 72 | } |
| 73 | } |
| 74 | } |