C#使用Protocol Buffer(ProtoBuf)进行对象的序列化与反序列化

2016-10-13 14:46 95人阅读 评论(0) 举报

分类： 文件操作（14）

Protocol Buffer是Google开发的数据格式,也是除了XML和JSON之外人气第三高的^^需要的朋友可以参考下

首先来说一下本文中例子所要实现的功能：

- 基于ProtoBuf序列化对象

下面来看具体的步骤：

Unity中使用ProtoBuf

导入DLL到Unity中，

创建网络传输的模型类：

```
[csharp]
using System;
using ProtoBuf;

//添加特性，表示可以被ProtoBuf工具序列化
[ProtoContract]
public class NetModel {
 //添加特性，表示该字段可以被序列化，1可以理解为下标
 [ProtoMember(1)]
 public int ID;
 [ProtoMember(2)]
 public string Commit;
 [ProtoMember(3)]
 public string Message;
}
```

using ProtoBuf;的引用需要使用protobuf-net.dll动态链接库文件

将这个DLL文件放入Plugins文件夹下就行

整个示例代码如下：

```
[csharp]
using UnityEngine;
using System;
using ProtoBuf;
using System.IO;
using System.Collections;

//添加特性，表示可以被ProtoBuf工具序列化
[ProtoContract]
public class NetModel {
    //添加特性，表示该字段可以被序列化，1可以理解为下标
    [ProtoMember(1)]
    public int ID;
    [ProtoMember(2)]
    public string Commit;
    [ProtoMember(3)]
    public string Message;
}

```

```
19.
20. public class ProtocolBuffer : MonoBehaviour {
21.
22.     // Use this for initialization
23.     void Start () {
24.         Test ();
25.     }
26.
27.     void Test()
28.     {
29.         //创建对象
30.         NetModel item = new NetModel(){ID = 1, Commit = "LanOu", Message = "Unity"};
31.         //序列化对象
32.         byte[] temp = Serialize(item);
33.         //ProtoBuf的优势一：小
34.         Debug.Log(temp.Length);
35.         //反序列化为对象
36.         NetModel result = DeSerialize(temp);
37.         Debug.Log(result.Message);
38.     }
39.
40.     // 将消息序列化为二进制的方法
41.     // < param name="model">要序列化的对象< /param>
42.     private byte[] Serialize(NetModel model)
43.     {
44.         try {
45.             //涉及格式转换，需要用到流，将二进制序列化到流中
46.             using (MemoryStream ms = new MemoryStream()) {
47.                 //使用ProtoBuf工具的序列化方法
48.                 ProtoBuf.Serializer.Serialize<NetModel> (ms, model);
49.                 //定义二级制数组，保存序列化后的结果
50.                 byte[] result = new byte[ms.Length];
51.                 //将流的位置设为0，起始点
52.                 ms.Position = 0;
53.                 //将流中的内容读取到二进制数组中
54.                 ms.Read (result, 0, result.Length);
55.                 return result;
56.             }
57.         } catch (Exception ex) {
58.             Debug.Log ("序列化失败: " + ex.ToString());
59.             return null;
60.         }
61.     }
62.
63.     // 将收到的消息反序列化成对象
64.     // < returns>The serialize.< /returns>
65.     // < param name="msg">收到的消息.</param>
66.     private NetModel DeSerialize(byte[] msg)
67.     {
68.         try {
69.             using (MemoryStream ms = new MemoryStream()) {
70.                 //将消息写入流中
71.                 ms.Write (msg, 0, msg.Length);
72.                 //将流的位置归0
73.                 ms.Position = 0;
74.                 //使用工具反序列化对象
75.                 NetModel result = ProtoBuf.Serializer.Deserialize<NetModel> (ms);
76.                 return result;
77.             }
78.         } catch (Exception ex) {
79.             Debug.Log("反序列化失败: " + ex.ToString());
80.             return null;
81.         }
82.     }
83. }
```