**Unity3D教程：解析Json**

Posted on 2013年07月10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 15 次

使用了LitJson这个开源组件，如果在C#脚本下使用该组件处理json很容易，在JavaScript脚本中处理其实也不难。首先要在Project里Import New Asset把LigJson.dll文件导入，然后在JavaScript脚本中就可以直接使用了。

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

|  |  |
| --- | --- |
| 1 | function Start() { |
| 2 | **var** s : **String** = "{'name':'aaa'}"; |
| 3 | **var** json : LitJson.JsonData = LitJson.JsonMapper.ToObject(s); |
| 4 | print(json["name"]); |
| 5 | } |
| 6 | function Update () { |
| 7 | } |

JsonData实现的接口如下：

IJsonWrapper, IList, ICollection, IEnumerable, IOrderedDictionary, IDictionary, IEquatable

JsonData需要从Json字符串返回的类型来判断该JsonData类型是否为数组，这一点可以从它的IsArray方法可以看出。

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

|  |  |
| --- | --- |
| 01 | **string** s = "[{'name':'lisi'},{'name':'zhangsan'}]"; |
| 02 | **string** a = "{'name':'aaa'}"; |
| 03 |  |
| 04 | JsonData[] jarr = JsonMapper.ToObject<JsonData[]>(s); |
| 05 | print(jarr. IsArray);*//返回true* |
| 06 | **if**(jarr.IsArray){ |
| 07 | **foreach**(JsonData jd **in** jarr){ |
| 08 | print(jd["name"]);*//返回结果: lisi* |
| 09 | *// zhangsan* |
| 10 | } |
| 11 | } |
| 12 | *//Unity3D教程手册：www.unitymanual.com* |
| 13 | JsonData jdata = JsonMapper.ToObject(s); |
| 14 | print(jdata.IsArray);*//返回true* |
| 15 | **if** (jdata.IsArray) |
| 16 | { |
| 17 | **for** (**int** i = 0; i < jdata.Count;i ++ ) |
| 18 | { |
| 19 | print(jdata[i]["name"]);*//返回的结果: lisi* |
| 20 | *// zhangsan* |
| 21 | } |
| 22 | } |
| 23 |  |
| 24 | JsonData j = JsonMapper.ToObject(a); |
| 25 | print(j.IsArray);*//返回false* |
| 26 | print(j["name"]);*//返回aaa* |

从上面的示例可以看出JsonData是否能以数组的方式访问需看json字符串返回的是否是数组类型的。示例展示了最基本得JsonData用法： 创建一个JsonData通常由JsonMapper的ToObject方法,这个方法有3个重载:

JsonMapper.ToObject(JsonReader reader);

JsonMapper.ToObject(string json);

JsonMapper.ToObject(System.IO.TextReader reader);

其中JsonReader类也是一个读取json字串的类，不过它是按行的方式读取，不能像key-value的方式读取json的值，第二个是最常见的创建JsonData的方式，它只需提供一个json字串即可,第三个是读取一个包含json字符的文本流来创建一个JsonData。

JsonMapper中还有一个ToObject方法:

public static T ToObject();

当然它也有三个重载，同上。目前我主要使用这个方法来创建JsonData类型的数组。

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

|  |  |
| --- | --- |
| 1 | JsonData[] jarr = JsonMapper.ToObject<JsonData[]>(s); |
| 2 | **foreach** (JsonData jd **in** jarr) |
| 3 | { |
| 4 | print(jd.IsArray);*// 返回false false* |
| 5 | } |

其实大家看JsonData类的这个方法就知道了:

public JsonData this[int index] { get; set; }

public JsonData this[string prop\_name] { get; set; }

注意，如果指定的key相同的话，后者将会覆盖

二：数组情况：

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

|  |  |
| --- | --- |
| 01 | JsonData j = JsonMapper.ToObject(s); |
| 02 |  |
| 03 | JsonData jdata1 = new JsonData(); |
| 04 |  |
| 05 | jdata1[“wwww”] = “wuwu”; |
| 06 |  |
| 07 | j.**Add**(jdata1); |
| 08 |  |
| 09 | print(JsonMapper.ToJson(j)); |
| 10 |  |
| 11 | *//返回: [{'name':'lisi'},{'name':'zhangsan'},{'wwww':'wuwu'}]* |