**xml文件的读取与保存**

Posted on 2013年04月19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 164 次

xml文件经常用在保存工程的信息：比如游戏进程中的位置信息，对抗双方的个人信息等。

使用xml文件：xml文件要以UTF-8的格式存储，但是这样做会使得programmer，可以从脚本中控制xml文件中的所有的字符，包括xml文件中的语法命令字符，因此会带来不安全隐患。

或者使用Unity3D的ISerializable类，它的好处是，可以将文件存成自己定义的后缀形式，并且2进制化存储，在Unity3D的帮助文档中有相关介绍。

代码如下：

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

|  |  |
| --- | --- |
| 001 | usingUnityEngine; |
| 002 | usingSystem.Collections; |
| 003 | usingSystem.Xml; |
| 004 | usingSystem.Xml.Serialization; |
| 005 | usingSystem.IO; |
| 006 | usingSystem.Text; |
| 007 |  |
| 008 | publicclass\_GameSaveLoad:MonoBehaviour{ |
| 009 |  |
| 010 | *// An example where the encoding can be found is at* |
| 011 | *// http://www.eggheadcafe.com/articles/system.xml.xmlserialization.asp* |
| 012 | *// We will just use the KISS method and cheat a little and use* |
| 013 | *// the examples from the web page since they are fully described* |
| 014 |  |
| 015 | *// This is our local private members* |
| 016 | Rect\_Save, \_Load, \_SaveMSG, \_LoadMSG; |
| 017 | bool\_ShouldSave, \_ShouldLoad,\_SwitchSave,\_SwitchLoad; |
| 018 | string\_FileLocation,\_FileName; |
| 019 | publicGameObject\_Player; |
| 020 | UserData myData; |
| 021 | string\_PlayerName; |
| 022 | string\_data; |
| 023 |  |
| 024 | Vector3VPosition; |
| 025 |  |
| 026 | *// When the EGO is instansiated the Start will trigger* |
| 027 | *// so we setup our initial values for our local members* |
| 028 | voidStart(){ |
| 029 | *// We setup our rectangles for our messages* |
| 030 | \_Save=newRect(10,80,100,20); |
| 031 | \_Load=newRect(10,100,100,20); |
| 032 | \_SaveMSG=newRect(10,120,400,40); |
| 033 | \_LoadMSG=newRect(10,140,400,40); |
| 034 |  |
| 035 | *// Where we want to save and load to and from* |
| 036 | \_FileLocation=Application.dataPath; |
| 037 | \_FileName="SaveData.xml"; |
| 038 |  |
| 039 | *// for now, lets just set the name to Joe Schmoe* |
| 040 | \_PlayerName ="Joe Schmoe"; |
| 041 |  |
| 042 | *// we need soemthing to store the information into* |
| 043 | myData=newUserData(); |
| 044 | } |
| 045 |  |
| 046 | voidUpdate(){} |
| 047 |  |
| 048 | voidOnGUI() |
| 049 | { |
| 050 |  |
| 051 | / |
| 052 | stringUTF8ByteArrayToString(**byte**[]characters) |
| 053 | { |
| 054 | UTF8Encoding encoding =newUTF8Encoding(); |
| 055 | stringconstructedString = encoding.GetString(characters); |
| 056 | **return**(constructedString); |
| 057 | } |
| 058 |  |
| 059 | **byte**[]StringToUTF8ByteArray(stringpXmlString) |
| 060 | { |
| 061 | UTF8Encoding encoding =newUTF8Encoding(); |
| 062 | **byte**[]byteArray = encoding.GetBytes(pXmlString); |
| 063 | returnbyteArray; |
| 064 | } |
| 065 |  |
| 066 | *// Here we serialize our UserData object of myData* |
| 067 | stringSerializeObject(objectpObject) |
| 068 | { |
| 069 | stringXmlizedString =**null**; |
| 070 | MemoryStream memoryStream =newMemoryStream(); |
| 071 | XmlSerializer xs =newXmlSerializer(typeof(UserData)); |
| 072 | XmlTextWriter xmlTextWriter =newXmlTextWriter(memoryStream, Encoding.UTF8); |
| 073 | xs.Serialize(xmlTextWriter, pObject); |
| 074 | memoryStream =(MemoryStream)xmlTextWriter.BaseStream; |
| 075 | XmlizedString = UTF8ByteArrayToString(memoryStream.ToArray()); |
| 076 | returnXmlizedString; |
| 077 | } |
| 078 |  |
| 079 | *// Here we deserialize it back into its original form* |
| 080 | objectDeserializeObject(stringpXmlizedString) |
| 081 | { |
| 082 | XmlSerializer xs =newXmlSerializer(typeof(UserData)); |
| 083 | MemoryStream memoryStream =newMemoryStream(StringToUTF8ByteArray(pXmlizedString)); |
| 084 | XmlTextWriter xmlTextWriter =newXmlTextWriter(memoryStream, Encoding.UTF8); |
| 085 | returnxs.Deserialize(memoryStream); |
| 086 | } |
| 087 |  |
| 088 | *// Finally our save and load methods for the file itself* |
| 089 | voidCreateXML() |
| 090 | { |
| 091 | StreamWriter writer; |
| 092 | FileInfo t =newFileInfo(\_FileLocation+"//"+ \_FileName); |
| 093 | **if**(!t.Exists) |
| 094 | { |
| 095 | writer = t.CreateText(); |
| 096 | } |
| 097 | **else** |
| 098 | { |
| 099 | t.Delete(); |
| 100 | writer = t.CreateText(); |
| 101 | } |
| 102 | writer.Write(\_data); |
| 103 | writer.Close(); |
| 104 | Debug.Log("File written."); |
| 105 | } |
| 106 |  |
| 107 | voidLoadXML() |
| 108 | { |
| 109 | StreamReader r = File.OpenText(\_FileLocation+"//"+ \_FileName); |
| 110 | string\_info = r.ReadToEnd(); |
| 111 | r.Close(); |
| 112 | \_data=\_info; |
| 113 | Debug.Log("File Read"); |
| 114 | } |
| 115 | } |
| 116 |  |
| 117 | *// UserData is our custom class that holds our defined objects we want to store in XML format* |
| 118 | publicclassUserData |
| 119 | { |
| 120 | *// We have to define a default instance of the structure* |
| 121 | publicDemoData \_iUser; |
| 122 | *// Default constructor doesn't really do anything at the moment* |
| 123 | publicUserData(){} |
| 124 |  |
| 125 | *// Anything we want to store in the XML file, we define it here* |
| 126 | publicstructDemoData |
| 127 | { |
| 128 | publicfloatx; |
| 129 | publicfloaty; |
| 130 | publicfloatz; |
| 131 | publicstringname; |
| 132 | } |
| 133 | } |
| 134 | 以下是javascript版本 |
| 135 | importSystem; |
| 136 | importSystem.Collections; |
| 137 | importSystem.Xml; |
| 138 | importSystem.Xml.Serialization; |
| 139 | importSystem.IO; |
| 140 | importSystem.Text; |
| 141 |  |
| 142 | *// Anything we want to store in the XML file, we define it here* |
| 143 | classDemoData |
| 144 | { |
| 145 | varx : **float**; |
| 146 | vary : **float**; |
| 147 | varz : **float**; |
| 148 | varname: **String**; |
| 149 | } |
| 150 |  |
| 151 | *// UserData is our custom class that holds our defined objects we want to store in XML format* |
| 152 | classUserData |
| 153 | { |
| 154 | *// We have to define a default instance of the structure* |
| 155 | publicvar\_iUser : DemoData =newDemoData(); |
| 156 | *// Default constructor doesn't really do anything at the moment* |
| 157 | functionUserData(){} |
| 158 | } |
| 159 |  |
| 160 | *//public class GameSaveLoad: MonoBehaviour {* |
| 161 |  |
| 162 | *// An example where the encoding can be found is at* |
| 163 | *// http://www.eggheadcafe.com/articles/system.xml.xmlserialization.asp* |
| 164 | *// We will just use the KISS method and cheat a little and use* |
| 165 | *// the examples from the web page since they are fully described* |
| 166 |  |
| 167 | *// This is our local private members* |
| 168 | privatevar\_Save : Rect; |
| 169 | privatevar\_Load : Rect; |
| 170 | privatevar\_SaveMSG : Rect; |
| 171 | privatevar\_LoadMSG : Rect; |
| 172 | *//var \_ShouldSave : boolean;* |
| 173 | *//var \_ShouldLoad : boolean;* |
| 174 | *//var \_SwitchSave : boolean;* |
| 175 | *//var \_SwitchLoad : boolean;* |
| 176 | privatevar\_FileLocation : **String**; |
| 177 | privatevar\_FileName : **String** ="SaveData.xml"; |
| 178 |  |
| 179 | *//public GameObject \_Player;* |
| 180 | var\_Player : GameObject; |
| 181 | var\_PlayerName : **String** ="Joe Schmoe"; |
| 182 |  |
| 183 | privatevarmyData : UserData; |
| 184 | privatevar\_data : **String**; |
| 185 |  |
| 186 | privatevarVPosition : Vector3; |
| 187 |  |
| 188 | *// When the EGO is instansiated the Start will trigger* |
| 189 | *// so we setup our initial values for our local members* |
| 190 | *//function Start () {* |
| 191 | functionAwake(){ |
| 192 | *// We setup our rectangles for our messages* |
| 193 | \_Save=newRect(10,80,100,20); |
| 194 | \_Load=newRect(10,100,100,20); |
| 195 | \_SaveMSG=newRect(10,120,200,40); |
| 196 | \_LoadMSG=newRect(10,140,200,40); |
| 197 |  |
| 198 | *// Where we want to save and load to and from* |
| 199 | \_FileLocation=Application.dataPath; |
| 200 |  |
| 201 |  |
| 202 | *// we need soemthing to store the information into* |
| 203 | myData=newUserData(); |
| 204 | } |
| 205 |  |
| 206 | functionUpdate(){} |
| 207 |  |
| 208 | functionOnGUI() |
| 209 | { |
| 210 |  |
| 211 | *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 212 | *// Loading The Player...* |
| 213 | *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 214 | **if**(GUI.Button(\_Load,"Load")){ |
| 215 |  |
| 216 | GUI.Label(\_LoadMSG,"Loading from: "+\_FileLocation); |
| 217 | *// Load our UserData into myData* |
| 218 | LoadXML(); |
| 219 | **if**(\_data.ToString()!="") |
| 220 | { |
| 221 | *// notice how I use a reference to type (UserData) here, you need this* |
| 222 | *// so that the returned object is converted into the correct type* |
| 223 | *//myData = (UserData)DeserializeObject(\_data);* |
| 224 | myData = DeserializeObject(\_data); |
| 225 | *// set the players position to the data we loaded* |
| 226 | VPosition=newVector3(myData.\_iUser.x,myData.\_iUser.y,myData.\_iUser.z); |
| 227 | \_Player.transform.position=VPosition; |
| 228 | *// just a way to show that we loaded in ok* |
| 229 | Debug.Log(myData.\_iUser.name); |
| 230 | } |
| 231 |  |
| 232 | } |
| 233 |  |
| 234 | *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 235 | *// Saving The Player...* |
| 236 | *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 237 | **if**(GUI.Button(\_Save,"Save")){ |
| 238 |  |
| 239 | GUI.Label(\_SaveMSG,"Saving to: "+\_FileLocation); |
| 240 | *//Debug.Log("SaveLoadXML: sanity check:"+ \_Player.transform.position.x);* |
| 241 |  |
| 242 | myData.\_iUser.x= \_Player.transform.position.x; |
| 243 | myData.\_iUser.y= \_Player.transform.position.y; |
| 244 | myData.\_iUser.z= \_Player.transform.position.z; |
| 245 | myData.\_iUser.name= \_PlayerName; |
| 246 |  |
| 247 | *// Time to creat our XML!* |
| 248 | \_data = SerializeObject(myData); |
| 249 | *// This is the final resulting XML from the serialization process* |
| 250 | CreateXML(); |
| 251 | Debug.Log(\_data); |
| 252 | } |
| 253 |  |
| 254 |  |
| 255 | } |
| 256 |  |
| 257 |  |
| 258 | *//string UTF8ByteArrayToString(byte[] characters)* |
| 259 | functionUTF8ByteArrayToString(characters : **byte**[]) |
| 260 | { |
| 261 | varencoding : UTF8Encoding =newUTF8Encoding(); |
| 262 | varconstructedString : **String** = encoding.GetString(characters); |
| 263 | **return**(constructedString); |
| 264 | } |
| 265 |  |
| 266 | *//byte[] StringToUTF8ByteArray(string pXmlString)* |
| 267 | functionStringToUTF8ByteArray(pXmlString : **String**) |
| 268 | { |
| 269 | varencoding : UTF8Encoding =newUTF8Encoding(); |
| 270 | varbyteArray : **byte**[]= encoding.GetBytes(pXmlString); |
| 271 | returnbyteArray; |
| 272 | } |
| 273 |  |
| 274 | *// Here we serialize our UserData object of myData* |
| 275 | *//string SerializeObject(object pObject)* |
| 276 | functionSerializeObject(pObject : **Object**) |
| 277 | { |
| 278 | varXmlizedString : **String** =**null**; |
| 279 | varmemoryStream : MemoryStream =newMemoryStream(); |
| 280 | varxs : XmlSerializer =newXmlSerializer(typeof(UserData)); |
| 281 | varxmlTextWriter : XmlTextWriter =newXmlTextWriter(memoryStream, Encoding.UTF8); |
| 282 | xs.Serialize(xmlTextWriter, pObject); |
| 283 | memoryStream = xmlTextWriter.BaseStream;*// (MemoryStream)* |
| 284 | XmlizedString = UTF8ByteArrayToString(memoryStream.ToArray()); |
| 285 | returnXmlizedString; |
| 286 | } |
| 287 |  |
| 288 | *// Here we deserialize it back into its original form* |
| 289 | *//object DeserializeObject(string pXmlizedString)* |
| 290 | functionDeserializeObject(pXmlizedString : **String**) |
| 291 | { |
| 292 | varxs : XmlSerializer =newXmlSerializer(typeof(UserData)); |
| 293 | varmemoryStream : MemoryStream =newMemoryStream(StringToUTF8ByteArray(pXmlizedString)); |
| 294 | varxmlTextWriter : XmlTextWriter =newXmlTextWriter(memoryStream, Encoding.UTF8); |
| 295 | returnxs.Deserialize(memoryStream); |
| 296 | } |
| 297 |  |
| 298 | *// Finally our save and load methods for the file itself* |
| 299 | functionCreateXML() |
| 300 | { |
| 301 | varwriter : StreamWriter; |
| 302 | *//FileInfo t = new FileInfo(\_FileLocation+"//"+ \_FileName);* |
| 303 | vart : FileInfo =newFileInfo(\_FileLocation+"/"+ \_FileName); |
| 304 | **if**(!t.Exists) |
| 305 | { |
| 306 | writer = t.CreateText(); |
| 307 | } |
| 308 | **else** |
| 309 | { |
| 310 | t.Delete(); |
| 311 | writer = t.CreateText(); |
| 312 | } |
| 313 | writer.Write(\_data); |
| 314 | writer.Close(); |
| 315 | Debug.Log("File written."); |
| 316 | } |
| 317 |  |
| 318 | functionLoadXML() |
| 319 | { |
| 320 | *//StreamReader r = File.OpenText(\_FileLocation+"//"+ \_FileName);* |
| 321 | varr : StreamReader = File.OpenText(\_FileLocation+"/"+ \_FileName); |
| 322 | var\_info : **String** = r.ReadToEnd(); |
| 323 | r.Close(); |
| 324 | \_data=\_info; |
| 325 | Debug.Log("File Read"); |
| 326 | } |
| 327 |  |
| 328 | *//}* |