**读写XML**

Posted on 2013年06月14日 by U3d / [Unity3D脚本/插件](http://www.unitymanual.com/category/script)/被围观 25 次

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：

方法1：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；

虽然Unity3D中的string类型说明中讲到保存的是unicode characters，但是实际上当xml文件比较大的时候，如果保存成unicode，就读不出来，如果保存成UTF-8就不存在这个问题；

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

|  |  |
| --- | --- |
| 001 | **using** UnityEngine; |
| 002 |  |
| 003 | **using** System.Collections; |
| 004 |  |
| 005 | **public** **class** ReadXML: MonoBehaviour { |
| 006 |  |
| 007 | *//store the read in file* |
| 008 |  |
| 009 | WWW statusFile; |
| 010 |  |
| 011 | *//decide wether the reading of xml has been finished* |
| 012 |  |
| 013 | **bool** isReadIn = **false**; |
| 014 |  |
| 015 | *// Use this for initialization* |
| 016 |  |
| 017 | IEnumeratorStart () {*//不能用void，否则没有办法使用yield* |
| 018 |  |
| 019 | isReadIn = **false**; |
| 020 |  |
| 021 | **yield** **return** StartCoroutine(ReadIn()); |
| 022 |  |
| 023 | isReadIn = **true**; |
| 024 |  |
| 025 | } |
| 026 |  |
| 027 | IEnumerator ReadIn() |
| 028 |  |
| 029 | { |
| 030 |  |
| 031 | **yield** **return** statusFile = new WWW(“[url=file:*///D:/unity/rotationAndcolor/Assets/information/testxml.xml%22);//]file:///D:/unity/rotationAndcolor/Assets/information/testxml.xml”);//[/url]注意路径的写法* |
| 032 |  |
| 033 | } |
| 034 |  |
| 035 | *// Update is called once per frame* |
| 036 |  |
| 037 | **void** Update () { |
| 038 |  |
| 039 | **if**(isReadIn) |
| 040 |  |
| 041 | { |
| 042 |  |
| 043 | **string** statusData = statusFile.data; |
| 044 |  |
| 045 | print(statusData.Length); |
| 046 |  |
| 047 | } |
| 048 |  |
| 049 | } |
| 050 |  |
| 051 | *//get the parameters in the xml file* |
| 052 |  |
| 053 | **void** getPatameters(**string** \_xmlString) |
| 054 |  |
| 055 | { |
| 056 |  |
| 057 | *//\_xmlString.[0]* |
| 058 |  |
| 059 | } |
| 060 |  |
| 061 | **void** postParameters() |
| 062 |  |
| 063 | { |
| 064 |  |
| 065 | } |
| 066 |  |
| 067 | }*//Unity3D教程手册* |
| 068 |  |
| 069 | B 这一段代码来自http:*//www.unifycommunity.com/wiki/index.php?title=Save\_and\_Load\_from\_XML* |
| 070 |  |
| 071 | usingUnityEngine; |
| 072 |  |
| 073 | usingSystem.Collections; |
| 074 |  |
| 075 | usingSystem.Xml; |
| 076 |  |
| 077 | usingSystem.Xml.Serialization; |
| 078 |  |
| 079 | usingSystem.IO; |
| 080 |  |
| 081 | usingSystem.Text; |
| 082 |  |
| 083 | publicclass\_GameSaveLoad:MonoBehaviour{ |
| 084 |  |
| 085 | *// An example where the encoding can be found is at* |
| 086 |  |
| 087 | *// http://www.eggheadcafe.com/articles/system.xml.xmlserialization.asp* |
| 088 |  |
| 089 | *// We will just use the KISS method and cheat a little and use* |
| 090 |  |
| 091 | *// the examples from the web page since they are fully described* |
| 092 |  |
| 093 | *// This is our local private members* |
| 094 |  |
| 095 | Rect\_Save, \_Load, \_SaveMSG, \_LoadMSG; |
| 096 |  |
| 097 | bool\_ShouldSave, \_ShouldLoad,\_SwitchSave,\_SwitchLoad; |
| 098 |  |
| 099 | string\_FileLocation,\_FileName; |
| 100 |  |
| 101 | publicGameObject\_Player; |
| 102 |  |
| 103 | UserData myData; |
| 104 |  |
| 105 | string\_PlayerName; |
| 106 |  |
| 107 | string\_data; |
| 108 |  |
| 109 | Vector3VPosition; |
| 110 |  |
| 111 | *// When the EGO is instansiated the Start will trigger* |
| 112 |  |
| 113 | *// so we setup our initial values for our local members* |
| 114 |  |
| 115 | voidStart(){ |
| 116 |  |
| 117 | *// We setup our rectangles for our messages* |
| 118 |  |
| 119 | \_Save=newRect(10,80,100,20); |
| 120 |  |
| 121 | \_Load=newRect(10,100,100,20); |
| 122 |  |
| 123 | \_SaveMSG=newRect(10,120,400,40); |
| 124 |  |
| 125 | \_LoadMSG=newRect(10,140,400,40); |
| 126 |  |
| 127 | *// Where we want to save and load to and from* |
| 128 |  |
| 129 | \_FileLocation=Application.dataPath; |
| 130 |  |
| 131 | \_FileName=“SaveData.xml”; |
| 132 |  |
| 133 | *// for now, lets just set the name to Joe Schmoe* |
| 134 |  |
| 135 | \_PlayerName =“Joe Schmoe”; |
| 136 |  |
| 137 | *// we need soemthing to store the information into* |
| 138 |  |
| 139 | myData=newUserData(); |
| 140 |  |
| 141 | } |
| 142 |  |
| 143 | voidUpdate(){} |
| 144 |  |
| 145 | voidOnGUI() |
| 146 |  |
| 147 | {<code lang=“csharp”>*//Unity3D教程手册* |
| 148 |  |
| 149 | *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 150 |  |
| 151 | *// Loading The Player...* |
| 152 |  |
| 153 | *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 154 |  |
| 155 | **if**(GUI.Button(\_Load,“Load”)){ |
| 156 |  |
| 157 | GUI.Label(\_LoadMSG,“Loading **from**: ”+\_FileLocation); |
| 158 |  |
| 159 | *// Load our UserData into myData* |
| 160 |  |
| 161 | LoadXML(); |
| 162 |  |
| 163 | **if**(\_data.ToString()!=“”) |
| 164 |  |
| 165 | { |
| 166 |  |
| 167 | *// notice how I use a reference to type (UserData) here, you need this* |
| 168 |  |
| 169 | *// so that the returned object is converted into the correct type* |
| 170 |  |
| 171 | myData =(UserData)DeserializeObject(\_data); |
| 172 |  |
| 173 | *// set the players position to the data we loaded* |
| 174 |  |
| 175 | VPosition=newVector3(myData.\_iUser.x,myData.\_iUser.y,myData.\_iUser.z); |
| 176 |  |
| 177 | \_Player.transform.position=VPosition; |
| 178 |  |
| 179 | *// just a way to show that we loaded in ok* |
| 180 |  |
| 181 | Debug.Log(myData.\_iUser.name); |
| 182 |  |
| 183 | } |
| 184 |  |
| 185 | } |
| 186 |  |
| 187 | *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 188 |  |
| 189 | *// Saving The Player...* |
| 190 |  |
| 191 | *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 192 |  |
| 193 | **if**(GUI.Button(\_Save,“Save”)){ |
| 194 |  |
| 195 | GUI.Label(\_SaveMSG,“Saving to: ”+\_FileLocation); |
| 196 |  |
| 197 | myData.\_iUser.x=\_Player.transform.position.x; |
| 198 |  |
| 199 | myData.\_iUser.y=\_Player.transform.position.y; |
| 200 |  |
| 201 | myData.\_iUser.z=\_Player.transform.position.z; |
| 202 |  |
| 203 | myData.\_iUser.name=\_PlayerName; |
| 204 |  |
| 205 | *// Time to creat our XML!* |
| 206 |  |
| 207 | \_data = SerializeObject(myData); |
| 208 |  |
| 209 | *// This is the final resulting XML from the serialization process* |
| 210 |  |
| 211 | CreateXML(); |
| 212 |  |
| 213 | Debug.Log(\_data); |
| 214 |  |
| 215 | } |
| 216 |  |
| 217 | } |
| 218 |  |
| 219 | */\* The following metods came from the referenced URL \*/* |
| 220 |  |
| 221 | stringUTF8ByteArrayToString(**byte**[]characters) |
| 222 |  |
| 223 | { |
| 224 |  |
| 225 | UTF8Encoding encoding =newUTF8Encoding(); |
| 226 |  |
| 227 | stringconstructedString = encoding.GetString(characters); |
| 228 |  |
| 229 | **return**(constructedString); |
| 230 |  |
| 231 | } |
| 232 |  |
| 233 | **byte**[]StringToUTF8ByteArray(stringpXmlString) |
| 234 |  |
| 235 | { |
| 236 |  |
| 237 | UTF8Encoding encoding =newUTF8Encoding(); |
| 238 |  |
| 239 | **byte**[]byteArray = encoding.GetBytes(pXmlString); |
| 240 |  |
| 241 | returnbyteArray; |
| 242 |  |
| 243 | } |
| 244 |  |
| 245 | *// Here we serialize our UserData object of myData* |
| 246 |  |
| 247 | stringSerializeObject(objectpObject) |
| 248 |  |
| 249 | { |
| 250 |  |
| 251 | stringXmlizedString =**null**; |
| 252 |  |
| 253 | MemoryStream memoryStream =newMemoryStream(); |
| 254 |  |
| 255 | XmlSerializer xs =newXmlSerializer(typeof(UserData)); |
| 256 |  |
| 257 | XmlTextWriter xmlTextWriter =newXmlTextWriter(memoryStream, Encoding.UTF8); |
| 258 |  |
| 259 | xs.Serialize(xmlTextWriter, pObject); |
| 260 |  |
| 261 | memoryStream =(MemoryStream)xmlTextWriter.BaseStream; |
| 262 |  |
| 263 | XmlizedString = UTF8ByteArrayToString(memoryStream.ToArray()); |
| 264 |  |
| 265 | returnXmlizedString; |
| 266 |  |
| 267 | } |
| 268 |  |
| 269 | *// Here we deserialize it back into its original form* |
| 270 |  |
| 271 | objectDeserializeObject(stringpXmlizedString) |
| 272 |  |
| 273 | { |
| 274 |  |
| 275 | XmlSerializer xs =newXmlSerializer(typeof(UserData)); |
| 276 |  |
| 277 | MemoryStream memoryStream =newMemoryStream(StringToUTF8ByteArray(pXmlizedString)); |
| 278 |  |
| 279 | XmlTextWriter xmlTextWriter =newXmlTextWriter(memoryStream, Encoding.UTF8); |
| 280 |  |
| 281 | returnxs.Deserialize(memoryStream); |
| 282 |  |
| 283 | } |
| 284 |  |
| 285 | *// Finally our save and load methods for the file itself* |
| 286 |  |
| 287 | voidCreateXML() |
| 288 |  |
| 289 | { |
| 290 |  |
| 291 | StreamWriter writer; |
| 292 |  |
| 293 | FileInfo t =newFileInfo(\_FileLocation+“*//”+ \_FileName);* |
| 294 |  |
| 295 | **if**(!t.Exists) |
| 296 |  |
| 297 | { |
| 298 |  |
| 299 | writer = t.CreateText(); |
| 300 |  |
| 301 | } |
| 302 |  |
| 303 | **else** |
| 304 |  |
| 305 | { |
| 306 |  |
| 307 | t.Delete(); |
| 308 |  |
| 309 | writer = t.CreateText(); |
| 310 |  |
| 311 | } |
| 312 |  |
| 313 | writer.Write(\_data); |
| 314 |  |
| 315 | writer.Close(); |
| 316 |  |
| 317 | Debug.Log(“File written.”); |
| 318 |  |
| 319 | } |
| 320 |  |
| 321 | voidLoadXML() |
| 322 |  |
| 323 | { |
| 324 |  |
| 325 | StreamReader r = File.OpenText(\_FileLocation+“*//”+ \_FileName);* |
| 326 |  |
| 327 | string\_info = r.ReadToEnd(); |
| 328 |  |
| 329 | r.Close(); |
| 330 |  |
| 331 | \_data=\_info; |
| 332 |  |
| 333 | Debug.Log(“File Read”); |
| 334 |  |
| 335 | } |
| 336 |  |
| 337 | } |
| 338 |  |
| 339 | *// UserData is our custom class that holds our defined objects we want to store in XML format* |
| 340 |  |
| 341 | publicclassUserData |
| 342 |  |
| 343 | { |
| 344 |  |
| 345 | *// We have to define a default instance of the structure* |
| 346 |  |
| 347 | publicDemoData \_iUser; |
| 348 |  |
| 349 | *// Default constructor doesn't really do anything at the moment* |
| 350 |  |
| 351 | publicUserData(){} |
| 352 |  |
| 353 | *// Anything we want to store in the XML file, we define it here* |
| 354 |  |
| 355 | publicstructDemoData |
| 356 |  |
| 357 | { |
| 358 |  |
| 359 | publicfloatx; |
| 360 |  |
| 361 | publicfloaty; |
| 362 |  |
| 363 | publicfloatz; |
| 364 |  |
| 365 | publicstringname; |
| 366 |  |
| 367 | } |
| 368 |  |
| 369 | } |
| 370 |  |
| 371 | 以下是javascript版本 |
| 372 |  |
| 373 | importSystem; |
| 374 |  |
| 375 | importSystem.Collections; |
| 376 |  |
| 377 | importSystem.Xml; |
| 378 |  |
| 379 | importSystem.Xml.Serialization; |
| 380 |  |
| 381 | importSystem.IO; |
| 382 |  |
| 383 | importSystem.Text; |
| 384 |  |
| 385 | *// Anything we want to store in the XML file, we define it here* |
| 386 |  |
| 387 | classDemoData |
| 388 |  |
| 389 | { |
| 390 |  |
| 391 | varx : **float**; |
| 392 |  |
| 393 | vary : **float**; |
| 394 |  |
| 395 | varz : **float**; |
| 396 |  |
| 397 | varname: **String**; |
| 398 |  |
| 399 | } |
| 400 |  |
| 401 | *// UserData is our custom class that holds our defined objects we want to store in XML format* |
| 402 |  |
| 403 | classUserData |
| 404 |  |
| 405 | { |
| 406 |  |
| 407 | *// We have to define a default instance of the structure* |
| 408 |  |
| 409 | publicvar\_iUser : DemoData =newDemoData(); |
| 410 |  |
| 411 | *// Default constructor doesn't really do anything at the moment* |
| 412 |  |
| 413 | functionUserData(){} |
| 414 |  |
| 415 | } |
| 416 |  |
| 417 | *//public class GameSaveLoad: MonoBehaviour {* |
| 418 |  |
| 419 | *// An example where the encoding can be found is at* |
| 420 |  |
| 421 | *// http://www.eggheadcafe.com/articles/system.xml.xmlserialization.asp* |
| 422 |  |
| 423 | *// We will just use the KISS method and cheat a little and use* |
| 424 |  |
| 425 | *// the examples from the web page since they are fully described* |
| 426 |  |
| 427 | *// This is our local private members* |
| 428 |  |
| 429 | privatevar\_Save : Rect; |
| 430 |  |
| 431 | privatevar\_Load : Rect; |
| 432 |  |
| 433 | privatevar\_SaveMSG : Rect; |
| 434 |  |
| 435 | privatevar\_LoadMSG : Rect; |
| 436 |  |
| 437 | *//var \_ShouldSave : boolean;* |
| 438 |  |
| 439 | *//var \_ShouldLoad : boolean;* |
| 440 |  |
| 441 | *//var \_SwitchSave : boolean;* |
| 442 |  |
| 443 | *//var \_SwitchLoad : boolean;* |
| 444 |  |
| 445 | privatevar\_FileLocation : **String**; |
| 446 |  |
| 447 | privatevar\_FileName : **String** =“SaveData.xml”; |
| 448 |  |
| 449 | *//public GameObject \_Player;* |
| 450 |  |
| 451 | var\_Player : GameObject; |
| 452 |  |
| 453 | var\_PlayerName : **String** =“Joe Schmoe”; |
| 454 |  |
| 455 | privatevarmyData : UserData; |
| 456 |  |
| 457 | privatevar\_data : **String**; |
| 458 |  |
| 459 | privatevarVPosition : Vector3; |
| 460 |  |
| 461 | *// When the EGO is instansiated the Start will trigger* |
| 462 |  |
| 463 | *// so we setup our initial values for our local members* |
| 464 |  |
| 465 | *//function Start () {* |
| 466 |  |
| 467 | functionAwake(){ |
| 468 |  |
| 469 | *// We setup our rectangles for our messages* |
| 470 |  |
| 471 | \_Save=newRect(10,80,100,20); |
| 472 |  |
| 473 | \_Load=newRect(10,100,100,20); |
| 474 |  |
| 475 | \_SaveMSG=newRect(10,120,200,40); |
| 476 |  |
| 477 | \_LoadMSG=newRect(10,140,200,40); |
| 478 |  |
| 479 | *// Where we want to save and load to and from* |
| 480 |  |
| 481 | \_FileLocation=Application.dataPath; |
| 482 |  |
| 483 | *// we need soemthing to store the information into* |
| 484 |  |
| 485 | myData=newUserData(); |
| 486 |  |
| 487 | } |
| 488 |  |
| 489 | functionUpdate(){} |
| 490 |  |
| 491 | functionOnGUI() |
| 492 |  |
| 493 | { |
| 494 |  |
| 495 | *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 496 |  |
| 497 | *// Loading The Player...* |
| 498 |  |
| 499 | *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 500 |  |
| 501 | 1 |
| 502 |  |
| 503 | GUI.Label(\_LoadMSG,“Loading **from**: ”+\_FileLocation); |
| 504 |  |
| 505 | *// Load our UserData into myData* |
| 506 |  |
| 507 | LoadXML(); |
| 508 |  |
| 509 | **if**(\_data.ToString()!=“”) |
| 510 |  |
| 511 | { |
| 512 |  |
| 513 | *// notice how I use a reference to type (UserData) here, you need this* |
| 514 |  |
| 515 | *// so that the returned object is converted into the correct type* |
| 516 |  |
| 517 | *//myData = (UserData)DeserializeObject(\_data);* |
| 518 |  |
| 519 | myData = DeserializeObject(\_data); |
| 520 |  |
| 521 | *// set the players position to the data we loaded* |
| 522 |  |
| 523 | VPosition=newVector3(myData.\_iUser.x,myData.\_iUser.y,myData.\_iUser.z); |
| 524 |  |
| 525 | \_Player.transform.position=VPosition; |
| 526 |  |
| 527 | *// just a way to show that we loaded in ok* |
| 528 |  |
| 529 | Debug.Log(myData.\_iUser.name); |
| 530 |  |
| 531 | } |
| 532 |  |
| 533 | } |
| 534 |  |
| 535 | *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 536 |  |
| 537 | *// Saving The Player...* |
| 538 |  |
| 539 | *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** |
| 540 |  |
| 541 | **if**(GUI.Button(\_Save,“Save”)){ |
| 542 |  |
| 543 | GUI.Label(\_SaveMSG,“Saving to: ”+\_FileLocation); |
| 544 |  |
| 545 | *//Debug.Log(“SaveLoadXML: sanity check:”+ \_Player.transform.position.x);* |
| 546 |  |
| 547 | myData.\_iUser.x= \_Player.transform.position.x; |
| 548 |  |
| 549 | myData.\_iUser.y= \_Player.transform.position.y; |
| 550 |  |
| 551 | myData.\_iUser.z= \_Player.transform.position.z; |
| 552 |  |
| 553 | myData.\_iUser.name= \_PlayerName; |
| 554 |  |
| 555 | *// Time to creat our XML!* |
| 556 |  |
| 557 | \_data = SerializeObject(myData); |
| 558 |  |
| 559 | *// This is the final resulting XML from the serialization process* |
| 560 |  |
| 561 | CreateXML(); |
| 562 |  |
| 563 | Debug.Log(\_data); |
| 564 |  |
| 565 | } |
| 566 |  |
| 567 | } |
| 568 |  |
| 569 | */\* The following metods came from the referenced URL \*/* |
| 570 |  |
| 571 | *//string UTF8ByteArrayToString(byte[] characters)* |
| 572 |  |
| 573 | functionUTF8ByteArrayToString(characters : **byte**[]) |
| 574 |  |
| 575 | { |
| 576 |  |
| 577 | varencoding : UTF8Encoding =newUTF8Encoding(); |
| 578 |  |
| 579 | varconstructedString : **String** = encoding.GetString(characters); |
| 580 |  |
| 581 | **return**(constructedString); |
| 582 |  |
| 583 | } |
| 584 |  |
| 585 | *//byte[] StringToUTF8ByteArray(string pXmlString)* |
| 586 |  |
| 587 | functionStringToUTF8ByteArray(pXmlString : **String**) |
| 588 |  |
| 589 | { |
| 590 |  |
| 591 | varencoding : UTF8Encoding =newUTF8Encoding(); |
| 592 |  |
| 593 | varbyteArray : **byte**[]= encoding.GetBytes(pXmlString); |
| 594 |  |
| 595 | returnbyteArray; |
| 596 |  |
| 597 | } |
| 598 |  |
| 599 | *// Here we serialize our UserData object of myData* |
| 600 |  |
| 601 | *//string SerializeObject(object pObject)* |
| 602 |  |
| 603 | functionSerializeObject(pObject : **Object**) |
| 604 |  |
| 605 | { |
| 606 |  |
| 607 | varXmlizedString : **String** =**null**; |
| 608 |  |
| 609 | varmemoryStream : MemoryStream =newMemoryStream(); |
| 610 |  |
| 611 | varxs : XmlSerializer =newXmlSerializer(typeof(UserData)); |
| 612 |  |
| 613 | varxmlTextWriter : XmlTextWriter =newXmlTextWriter(memoryStream, Encoding.UTF8); |
| 614 |  |
| 615 | xs.Serialize(xmlTextWriter, pObject); |
| 616 |  |
| 617 | memoryStream = xmlTextWriter.BaseStream;*// (MemoryStream)* |
| 618 |  |
| 619 | XmlizedString = UTF8ByteArrayToString(memoryStream.ToArray()); |
| 620 |  |
| 621 | returnXmlizedString; |
| 622 |  |
| 623 | } |
| 624 |  |
| 625 | *// Here we deserialize it back into its original form* |
| 626 |  |
| 627 | *//object DeserializeObject(string pXmlizedString)* |
| 628 |  |
| 629 | functionDeserializeObject(pXmlizedString : **String**) |
| 630 |  |
| 631 | { |
| 632 |  |
| 633 | varxs : XmlSerializer =newXmlSerializer(typeof(UserData)); |
| 634 |  |
| 635 | varmemoryStream : MemoryStream =newMemoryStream(StringToUTF8ByteArray(pXmlizedString)); |
| 636 |  |
| 637 | varxmlTextWriter : XmlTextWriter =newXmlTextWriter(memoryStream, Encoding.UTF8); |
| 638 |  |
| 639 | returnxs.Deserialize(memoryStream); |
| 640 |  |
| 641 | } |
| 642 |  |
| 643 | *// Finally our save and load methods for the file itself* |
| 644 |  |
| 645 | functionCreateXML() |
| 646 |  |
| 647 | { |
| 648 |  |
| 649 | varwriter : StreamWriter; |
| 650 |  |
| 651 | *//FileInfo t = new FileInfo(\_FileLocation+“//”+ \_FileName);* |
| 652 |  |
| 653 | vart : FileInfo =newFileInfo(\_FileLocation+“/”+ \_FileName); |
| 654 |  |
| 655 | **if**(!t.Exists) |
| 656 |  |
| 657 | { |
| 658 |  |
| 659 | writer = t.CreateText(); |
| 660 |  |
| 661 | } |
| 662 |  |
| 663 | **else** |
| 664 |  |
| 665 | { |
| 666 |  |
| 667 | t.Delete(); |
| 668 |  |
| 669 | writer = t.CreateText(); |
| 670 |  |
| 671 | } |
| 672 |  |
| 673 | writer.Write(\_data); |
| 674 |  |
| 675 | writer.Close(); |
| 676 |  |
| 677 | Debug.Log(“File written.”); |
| 678 |  |
| 679 | } |
| 680 |  |
| 681 | functionLoadXML() |
| 682 |  |
| 683 | { |
| 684 |  |
| 685 | *//StreamReader r = File.OpenText(\_FileLocation+“//”+ \_FileName);* |
| 686 |  |
| 687 | varr : StreamReader = File.OpenText(\_FileLocation+“/”+ \_FileName); |
| 688 |  |
| 689 | var\_info : **String** = r.ReadToEnd(); |
| 690 |  |
| 691 | r.Close(); |
| 692 |  |
| 693 | \_data=\_info; |
| 694 |  |
| 695 | Debug.Log(“File Read”); |
| 696 |  |
| 697 | } |
| 698 |  |
| 699 | *//}* |