神杀国战自定义场景教程

在即将发布的国战新版本中，Lua已经有创建一个属于自己的场景（例如绝经之战）的能力。在本教程中，我（Xusine）将向广大Luaer介绍一下创建自定义场景的具体内容。

# 概述

我们先来看一下相关函数的定义：

**function** sgs.CreateLuaScenario**(**spec**)**

**assert(type(**spec.name**)** **==** "string"**)**

**assert(type(**spec.rule **==** "userdata"**)** **and** spec.rule**:**inherits**(**"LuaTriggerSkill"**))**

**assert(type(**spec.player\_count**)** **==** "number"**)**

**assert(type(**spec.random\_seat**)** **==** "boolean"**)**

**local** scenario **=** sgs.LuaScenario**(**spec.name**)**

scenario**:**setRandomSeat**(**spec.random\_seat**)**

scenario**:**setRule**(**spec.rule**)**

**if** **type(**spec.expose\_role**)** **==** "boolean" **then**

scenario.expose\_role **=** spec.expose\_role

**end**

scenario.player\_count **=** spec.player\_count

**if** **type(**spec.on\_assign**)** **==** "function" **then**

scenario.general\_selection **=** **false**

scenario.on\_assign **=** **function(**self**,**room**)**

**local** general1**,**general2**,**kingdom **=** spec.on\_assign**(**self**,**room**)**

**return** **table.concat(**general1**,**"+"**),table.concat(**general2**,**"+"**),table.concat(**kingdom**,**"+"**)**

**end**

**else**

scenario.general\_selection **=** **true**

**end**

scenario.relation **=** spec.relation **or** 0

scenario.on\_tag\_set **=** spec.on\_tag\_set **or** 0

**return** scenario

**end**

这是从sgs\_ex.lua中复制的相关代码。

从这段代码中我们可以看到，要使用函数sgs. CreateLuaScenario来创建一个场景，需要这样的几个参数：

1.name

字符串类型，一个场景的名字。

需要在翻译表里面翻译一下，要不然就显示英语字母了。

2.rule

LuaTriggerSkill类型，即一个Lua触发技。该技能代表的是场景规则。（比如在绝经之战游戏开始时获得随即技能）该技能仅需要写name、events、on\_effect三个成员即可。

3. player\_count

整数类型，该场景的玩家数目。

不能大于10.

4. expose\_role

布尔类型，是否暴露身份。（在国战中就是所谓的国籍了）

Xusine提示:这个值在目前还是有点小问题，我会尽快修复。

5.random\_seat

布尔类型，是否随机位置。

6. on\_assign

Lua函数，分配武将。

如果不填写这个函数，那么就按照默认的分配武将的形势分配。

对应某些特定场景（比如说守卫剑阁），需要来手动分配武将。下面我们以源码为例，来分析一下这个函数。

void JiangeDefenseScenario**::**assign**(**QStringList **&**generals**,** QStringList **&**generals2**,** QStringList **&**kingdoms**,** Room **\***room**)** const

**{**

QMap**<**QString**,** QStringList**>** roles**;**

QStringList wei\_roles**,** shu\_roles**;**

wei\_roles **<<** "ghost" **<<** "machine" **<<** "human" **<<** "human"**;**

shu\_roles **<<** "ghost" **<<** "machine" **<<** "human" **<<** "human"**;**

roles**.**insert**(**"wei"**,** wei\_roles**);**

roles**.**insert**(**"shu"**,** shu\_roles**);**

qShuffle**(**kingdoms**);**

QStringList wei\_generals**,** shu\_generals**;**

**foreach** **(**const QString **&**general**,** Sanguosha**->**getLimitedGeneralNames**())** **{**

**if** **(**general**.**startsWith**(**"lord\_"**))** **continue;**

**if** **(**BanPair**::**isBanned**(**general**))** **continue;**

QString kingdom **=** Sanguosha**->**getGeneral**(**general**)->**getKingdom**();**

**if** **(**kingdom **==** "wei"**)**

wei\_generals **<<** general**;**

**else** **if** **(**kingdom **==** "shu"**)**

shu\_generals **<<** general**;**

**}**

qShuffle**(**wei\_generals**);**

qShuffle**(**shu\_generals**);**

**Q\_ASSERT(**wei\_generals**.**length**()** **>=** 10 **&&** shu\_generals**.**length**()** **>=** 10**);**

QMap**<**ServerPlayer **\*,**QString**>** human\_map**;** // Rara said, human couldn't get ghost or machine as its general.

QList**<**ServerPlayer **\*>** players **=** room**->**getPlayers**();**

**for** **(**int i **=** 0**;** i **<** 8**;** i**++)** **{**

**if** **(**players**[**i**]->**getState**()** **==** "online"**)** **{**

QStringList choices**;**

**foreach(**const QString **&**kingdom**,** roles**.**keys**())**

**if** **(**roles**[**kingdom**].**contains**(**"human"**))**

choices **<<** kingdom**;**

QString choice **=** choices**.**at**(**qrand**()** **%** choices**.**length**());**

QStringList role\_list **=** roles**[**choice**];**

role\_list**.**removeOne**(**"human"**);**

roles**[**choice**]** **=** role\_list**;**

**if** **(**choice **==** "wei"**)** **{**

**for** **(**int j **=** 0**;** j **<** 5**;** j**++)**

players**[**i**]->**addToSelected**(**wei\_generals**.**takeFirst**());**

human\_map**.**insert**(**players**[**i**],** "wei"**);**

**}** **else** **if** **(**choice **==** "shu"**)** **{**

**for** **(**int j **=** 0**;** j **<** 5**;** j**++)**

players**[**i**]->**addToSelected**(**shu\_generals**.**takeFirst**());**

human\_map**.**insert**(**players**[**i**],** "shu"**);**

**}**

**}**

**}**

QList**<**ServerPlayer **\*>** humans **=** human\_map**.**keys**();**

room**->**chooseGenerals**(**humans**,** **true,** **true);**

**for** **(**int i **=** 0**;** i **<** 8**;** i**++)** **{**

**if** **(**human\_map**.**contains**(**players**[**i**]))** **{**

kingdoms **<<** human\_map**[**players**[**i**]];**

generals **<<** players**[**i**]->**getGeneralName**();**

generals2 **<<** players**[**i**]->**getGeneral2Name**();**

**}** **else** **{**

QStringList kingdom\_choices**;**

**foreach(**const QString **&**kingdom**,** roles**.**keys**())**

**if** **(!**roles**[**kingdom**].**isEmpty**())**

kingdom\_choices **<<** kingdom**;**

QString kingdom **=** kingdom\_choices**.**at**(**qrand**()** **%** kingdom\_choices**.**length**());**

kingdoms **<<** kingdom**;**

QStringList role\_list **=** roles**[**kingdom**];**

QString role **=** role\_list**.**at**(**qrand**()** **%** role\_list**.**length**());**

role\_list**.**removeOne**(**role**);**

roles**[**kingdom**]** **=** role\_list**;**

**if** **(**role **==** "ghost"**)** **{**

QString name **=** kingdom **==** "wei" **?** getRandomWeiGhost**()** **:** getRandomShuGhost**();**

generals **<<** name**;**

generals2 **<<** name**;**

**}** **else** **if** **(**role **==** "machine"**)** **{**

QString name **=** kingdom **==** "wei" **?** getRandomWeiMachine**()** **:** getRandomShuMachine**();**

generals **<<** name**;**

generals2 **<<** name**;**

**}** **else** **if** **(**role **==** "human"**)** **{**

int n **=** kingdom **==** "wei" **?** 4 **:** 5**;**

QStringList choices**;**

**for** **(**int j **=** 0**;** j **<** n**;** j**++)**

choices **<<** **((**kingdom **==** "wei"**)** **?** wei\_generals**.**takeFirst**()** **:** shu\_generals**.**takeFirst**());**

QString answer **=** room**->**askForGeneral**(**players**[**i**],** choices**,** QString**(),** **false);**

generals **<<** answer**.**split**(**"+"**).**first**();**

generals2 **<<** answer**.**split**(**"+"**).**last**();**

**}**

**}**

**}**

**}**

该函数有两个参数，即场景本身和所在的Room。

源码使用的是参数传递地址来作为返回值，我们Lua可以直接返回3个table。

第一个table中保存的是从1号位到最后一号位的主将的objectName（比如说liubei、sunquan）。

第二个table中保存的是从1号位到最后一号位的副将的objectName（比如说guanyu、sunshangxiang）。

第三个table中保存的是从1号位到最后一号位的国籍。（比如说wei、shu）

总之，这三个table中的值都是字符串，否则会出现各种错误。附录A将介绍一个更加有力的选择武将的方法。

7. relation

Lua函数，参数有三个，场景本身、一名玩家A和另一名玩家B。其中的玩家A和B都可以获得Room。

这个函数的主要作用是矫正电脑之间的AI关系。

如果玩家A和玩家B为盟友，返回sgs. AI\_Friend

同理，如果为敌军，则返回sgs. AI\_Enemy

中立的话要返回sgs. AI\_Neutrality

Xusine提示：在目前的国战AI中没有综合考虑该函数，因此请尽量在不改变国战对抗规则的基础上制作场景。

8. on\_tag\_set

Lua函数，参数有三个，场景本身、场景所在的Room、Tag的name，无返回值。

这个函数的主要是在设置Tag（仅限于在Room中设置Tag）的时候触发的。可以用Room::getTag来获得QVariant。

可以看出，如果操作好这三个函数，我们将很容易的写出完美的场景。

# 实例

我们来制作名字叫做一统天下的场景。在这个场景中，四个势力都登场，每个势力有两名角色，位置随机。每个势力的角色死亡后可以选择另外2名未使用过的该势力武将作为自己的武将登场，每一个势力限三次。

相关文件请参考本文档附带的Xmode.lua

分析：先建立这样一个table来存放场景数据。

Xmode **=** **{**

name **=** "Xmode\_hegemony"**,**

expose\_role **=** **false,**

player\_count **=** 8**,**

random\_seat **=** **true,**

rule **=** XmodeRule**,**

同时，我们还要写翻译表。

sgs.LoadTranslationTable**{**

**[**"Xmode\_hegemony"**]** **=** "一统天下"**,**

**}**

这些大家都能看懂的。（那个Xmode的Rule我们一会再介绍）

同时，我们定义几个函数，以方便我们下面的代码工作。

**local** hash **=** **{**wei **=** **{},**shu **=** **{},**wu **=** **{},**qun **=** **{}}**

**for** \_**,**general **in** **pairs(**sgs.Sanguosha**:**getLimitedGeneralNames**())do**

**if** general**:**startsWith**(**"lord\_"**)** **then** continue **end**

**if** hash**[**sgs.Sanguosha**:**getGeneral**(**general**):**getKingdom**()]** **then**

**table.insert(**hash**[**sgs.Sanguosha**:**getGeneral**(**general**):**getKingdom**()],**general**)**

**end**

**end**

**function** getRandomGenerals **(**n**,**kingdom**,**exceptions**)**

hash**[**kingdom**]** **=** table.Shuffle**(**hash**[**kingdom**])**

**local** result **=** **{}**

**for** \_**,**general **in** **pairs(**hash**[**kingdom**])do**

**if** **#**result **==** n **then** **break** **end**

**if** **not** table.contains**(**exceptions**,**general**)** **then**

**table.insert(**result**,**general**)**

**end**

**end**

**return** result

**end**

然后我们在来处理Rule。基本上就是参考飞龙夺冠的设计：

XmodeRule **=** sgs.CreateTriggerSkill**{**

name **=** "XmodeRule"**,**

events **=** **{**sgs.BuryVictim**},**

on\_effect **=** **function(**self**,** evnet**,** room**,** player**,** data**,**ask\_who**)**

**local** times **=** room**:**getTag**(**player**:**getKingdom**()..**"\_Change"**):**toInt**()**

player**:**speak**(**times**)**

**if** times **==** 2 **then** **return** **false** **end**

**local** used **=** room**:**getTag**(**"Xmode\_UsedGeneral"**):**toString**():**split**(**"+"**)**

**local** random\_general **=** getRandomGenerals**(**sgs.GetConfig**(**"HegemonyMaxChoice"**,**0**),**player**:**getKingdom**(),**used**)**

**local** choice **=** room**:**askForGeneral**(**player**,table.concat(**random\_general**,**"+"**),nil,false):**split**(**"+"**)**

table.insertTable**(**used**,**choice**)**

room**:**setTag**(**"Xmode\_UsedGeneral"**,**sgs.QVariant**(table.concat(**used**,**"+"**)))**

room**:**doDragonPhoenix**(**player**,**choice**[**1**],** choice**[**2**],true,**player**:**getKingdom**(),false,**""**,true)**

player**:**drawCards**(**4**)**

times **=** times **+** 1

room**:**setTag**(**player**:**getKingdom**()..**"\_Change"**,**sgs.QVariant**(**times**))**

**return** **true** --不知道能不能处理飞龙

**end,**

**}**

如果你看不懂那个doDragonPhoenix函数的使用方法，我们将在附录A中介绍这个函数的使用方式。

当然，更重要的是on\_assign部分。我们在使用正常模式的选将的时候，是多个玩家同时进行选将，而不是单一的askForGeneral的询问。因此，我们将使用强有力的chooseGenral函数来多线程同时选择武将。

on\_assign **=** **function(**self**,** room**)**

**local** generals**,** generals2**,** kingdoms **=** **{},{},{}**

**local** kingdom **=** **{**"wei"**,**"shu"**,**"wu"**,**"qun"**,}**

**local** rules\_count **=** **{[**"wei"**]** **=** 0**,[**"shu"**]** **=** 0**,[**"wu"**]** **=** 0**,[**"qun"**]** **=** 0**}**

**for** i **=** 1**,** 8**,** 1 **do**

**local** role **=** kingdom**[math.random(**1**,#**kingdom**)]**

rules\_count**[**role**]** **=** rules\_count**[**role**]** **+** 1

**if** rules\_count**[**role**]** **==** 2 **then** table.removeOne**(**kingdom**,**role**)** **end**

**table.insert(**kingdoms**,** role**)**

**end**

--上面的代码是为了随机分配国家，每个势力有两个人。

--相信大家都能看懂。

**local** selected **=** **{}**

**for** i **=** 1**,**8**,**1 **do**

**local** player **=** room**:**getPlayers**():**at**(**i**-**1**)**

player**:**clearSelected**()**

**local** random\_general **=** getRandomGenerals**(**sgs.GetConfig**(**"HegemonyMaxChoice"**,**0**),**kingdoms**[**i**],**selected**)**

**for** \_**,**general **in** **pairs(**random\_general**)do**

player**:**addToSelected**(**general**)**

**table.insert(**selected**,**general**)**

**end**

**end**

room**:**chooseGenerals**(**room**:**getPlayers**(),true,true)**

--这部分将在附录A中介绍。

**for** i **=** 1**,**8**,**1 **do** --依次设置genera1、general2。

**local** player **=** room**:**getPlayers**():**at**(**i**-**1**)**

generals**[**i**]** **=** player**:**getGeneralName**()**

generals2**[**i**]** **=** player**:**getGeneral2Name**()**

**local** used **=** room**:**getTag**(**"Xmode\_UsedGeneral"**):**toString**():**split**(**"+"**)**

**table.insert(**used**,**generals**[**i**])**

**table.insert(**used**,**generals2**[**i**])**

room**:**setTag**(**"Xmode\_UsedGeneral"**,**sgs.QVariant**(table.concat(**used**,**"+"**)))** --记录使用过的武将。

**end**

**return** generals**,** generals2**,** kingdoms

**end,**

以上代码相信大家都能看懂。

最后我们需要的是返回这个场景啦：

**return** sgs.CreateLuaScenario**(**Xmode**)**

至此，我们的场景制作完成。

# 附录A 几个特殊函数的介绍

### table.Shuffle

这个函数的作用是将一个table顺序打乱并返回，我们直接看定义好了。

**function** table.Shuffle**(**list**)**

**local** result **=** **{}**

**while** **#**list **>** 0 **do**

**local** value **=** list**[math.random(**1**,#**list**)]**

**table.insert(**result**,**value**)**

table.removeOne**(**list**,**value**)**

**end**

**return** result

**end**

因此，实例中是这样使用的：

hash**[**kingdom**]** **=** table.Shuffle**(**hash**[**kingdom**])**

### doDragonPhoenix

从字面上看，这个函数是执行飞龙夺冠的效果的。

函数声明如下：

void doDragonPhoenix**(**ServerPlayer **\***target**,** const char **\***general1\_name**,** const char **\***general2\_name**,** bool full\_state **=** **true,**const char **\***kingdom **=** QString**(),** bool sendLog **=** **true,** const char **\***show\_flags **=** QString**(),** bool resetHp **=** **false);**

Xusine提示：使用这个函数时务必小心，可能会有不可思议的bug发生。如果发现Bug，请及时反馈给开发组。

参数含义如下：

target:被操作的玩家，务必是死亡玩家。

general1\_name:替换的主将的name

general2\_name:替换的副将的name

full\_state:是否满状态复活。（如果true的话会给限定Mark并重置武将牌）

kingdom:target的国籍。不填的话默认使用主将的国籍。

sendLog:是否发送变身为XX武将的Log

show\_flags:替换武将后的亮将标记，如果包含字符“h”的话就明亮主将，如果包括“d”字符的话就明亮副将。

ResetHp:是否重新计算MaxHp和Hp，默认为不重新计算。

我们的实例是这样使用的：

room**:**doDragonPhoenix**(**player**,**choice**[**1**],** choice**[**2**],true,**player**:**getKingdom**(),false,**""**,true)**

对比之下，相信大家能够看懂。

### chooseGenerals

这个是多线程的选将函数，在使用之间请为玩家分发待选择的武将。

分配的方式可以参考实例中的这部分：

**for** i **=** 1**,**8**,**1 **do** --开始给每名玩家分配待选择的武将。

**local** player **=** room**:**getPlayers**():**at**(**i**-**1**)** --获取相关玩家

player**:**clearSelected**()** --清除已经分配的武将

--如果不清除的话可能会获得上次askForGeneral的武将。

**local** random\_general **=** getRandomGenerals**(**sgs.GetConfig**(**"HegemonyMaxChoice"**,**0**),**kingdoms**[**i**],**selected**)**

--随机获得HegemonyMaxChoice个武将。

--函数getRandomGenerals在本文件内定义，可以参考之。

**for** \_**,**general **in** **pairs(**random\_general**)do**

player**:**addToSelected**(**general**)**

--这个函数就是把武将分发给相关玩家。

--分发的武将会在选将框中出现。

**table.insert(**selected**,**general**)**

**end**

**end**

在完成这些工作后，我们来看一下这个函数的声明。

void chooseGenerals**(**QList**<**ServerPlayer **\*>** **&**assign\_players**,**bool has\_assign **=** **false,**bool is\_scenario **=** **false);**

Xusine提示：这个函数仅仅仅仅仅仅仅仅能在设计场景的on\_assign函数中使用，否则后果自负。

参数如下：

Assign\_players：待分配武将的玩家，没有默认值。（在on\_assign中应当为room:getPlayers()，即所有玩家都分配）

Has\_assign：是否已经手动分发待选择的武将。如果你希望系统来给你分配武将的话，可以填写false或者不填，否则应当填写true。

Is\_scenario：是否为场景模式。如果为场景模式的话，将不会给玩家设置好武将。（如果在assign函数中调用的话请填写true）

这个函数的结果是给每名玩家设置好武将，可以用player:getGeneralName()和player: getGeneral2Name()获取到。

对于场景来说，我们还需要这样设置，即给on\_assign函数返回值中的general1和general2的两个table来设置值。实例中是这样实现的：

**for** i **=** 1**,**8**,**1 **do** --依次设置genera1、general2。

**local** player **=** room**:**getPlayers**():**at**(**i**-**1**)**

generals**[**i**]** **=** player**:**getGeneralName**()** --获取武将，这个是chooseGenerals分配的。

generals2**[**i**]** **=** player**:**getGeneral2Name**()** --同上

**local** used **=** room**:**getTag**(**"Xmode\_UsedGeneral"**):**toString**():**split**(**"+"**)**

**table.insert(**used**,**generals**[**i**])** --设置返回值generals。

**table.insert(**used**,**generals2**[**i**])** --设置返回值generals2。

room**:**setTag**(**"Xmode\_UsedGeneral"**,**sgs.QVariant**(table.concat(**used**,**"+"**)))** --记录使用过的武将。

**end**