**转码工具**

基于cs2lua开源方案，快速把C#代码转换为Lua脚本。

**目录**

/bin/cs2lua\_format\_xlua.py 转码脚本

/bin/sign.py 签名脚本

/bin/core lua框架脚本。

/script 原始脚本

/output 输出脚本

**流程 (转码->签名)**

1. 修复代码已提交到主线。
2. 工具目录/config.ini，配置自己的主线工程目录和发布目录。

2、运行run.bat –cs2lua ===> 工程目录/lua。

3、拷贝lua scripts ===> 工具目录/script。

4、运行run.bat –xlua ===> 工具目录/output/unsigned。

5、运行run.bat –sign ===> 工具目录/output/signed。

6、运行run.bat –push ===> 发布目录。

**1.-cs2lua**

使用cs2lua，把C#代码转换为Lua脚本。

**2. –xlua**

把cs2lua脚本转换为热更脚本。

解决的需求：

1、原始脚本，存在很多特定结构，和cs2lua框架本身存在耦合，通过这个脚本，提取出通用的逻辑部分，只供热更新流程使用。

2、添加上自己项目的的接口。

3、一定程度转换写法，更偏向于xlua的风格，使编码更符合xlua的特性和原理。

热更脚本约定了通用的格式，流程上只是把对应的逻辑部分塞进去即可。

所以该工具只是帮你更快地转换脚本，省去重复的手头工作，同样的修复点，参照原来C#代码，你也可以写出相似的热更脚本。

**但目前该工具还不完善，建议使用该工具转换出来后，自己检查和整理一下。**

**3.-sign**

Xlua提供的FilesSignature签名机制，主要保证热更脚本传输前后的一致性。

**4. –push**

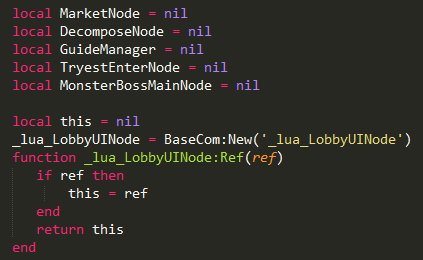
发布。

**热更脚本**

一个c#文件对应一个热更脚本，每个脚本都有通用的格式。

以” LobbyUINode.lua”为例：

**声明部分**



1. 模块名

\_lua\_LobbyUINode Lua对象：self

1. C#引用（this）

由\_lua\_LobbyUINode:Ref(ref)获取。

后续通过This.XXX、this:XXX()进行调用。

**热更函数列表**

