# Lua开发规范

By: pengwan

Version : 1.0.0.0

Update: chal (2013-6-28)

Info：create

Version : 1.0.0.1

Update: chal (2013-7-11)

Info：add module struct

Version : 2.0.0.0

Update: chal (2015-9-6)

Info：new standard

## 一 目录/文件/函数/变量命名规范

[1] 全小写英文

[2] 单词以“\_”连接

## 二 类命名规范

[1] 全小写英文

[2] 命名规则为：”目录名\_文件名”

如Script/player/xinfa.lua , 则类名为 player\_xinfa

## 三 配置文件规范

[1] 统一放置在game\_config 目录下（之前的项目命名可能有所不同）

[2] 配置表的父表设置为 game\_config（之前的项目命名可能有所不同）

[3] 配置表名与文件名相同，并且一个配置文件中只在game\_config下注册与文件名一

致的一张子表。

如： Script/game\_config/xinfa\_cfg.lua 则内容为：

game\_config. xinfa\_cfg = {

sub\_cfg1 = {},

sub\_cfg2 = {},

}

[4] 配置表默认为只读（不可修改），对于奖励类等敏感配置不可修改只读属性；其他 配置如果功能内需要修改则可以设置非只读标识。

## 四 编码格式规范



推荐使用RPC，不再使用消息号的形式

所有文件都采用dofile进行加载，不再需要在文件末尾return

## 五 特殊规范

### 1 与C交互的接口规范

[1] 定义为全局函数

[2] 命名规范：

* C提供给lua的接口： “ci\_funciton\_a”
* Lua提供给C的接口： ”si\_funciton\_a”
* Lua 对C提供的接口的封装： ”pi\_function\_a”

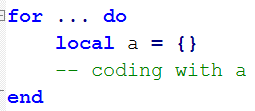
## 六 性能规范

**详见文档“lua优化准则.xlsx”**

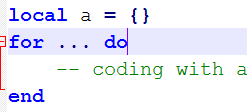
### 1 对于C++和lua的库函数在文件中用local做上值声明，对于脚本中的全局函数不用做上值引用

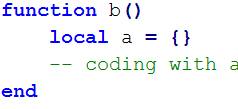
### 2 复用原则(table)

尽量在更上层的闭包内定义。

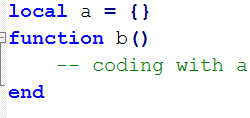


Better：





Better if call frequently:



## 七 开发技巧总结

1 函数性能跟踪

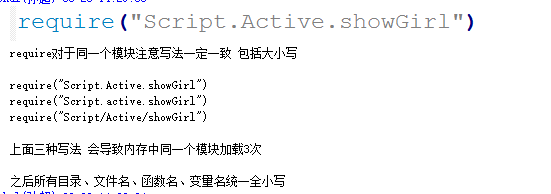
TRACE\_BEGIN()

…(函数调用)

Local spend\_time = TRACE\_END()

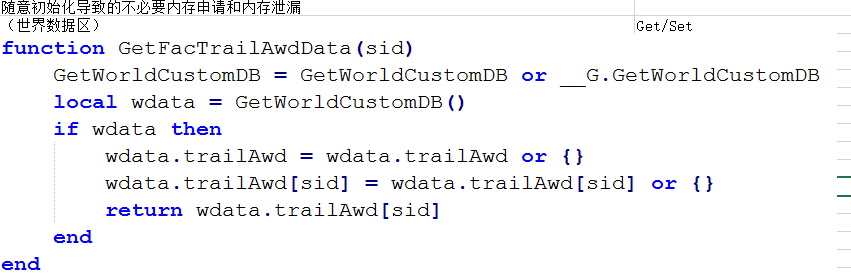
spend\_time 为函数调用消耗的毫秒数

2申请数据时不再指定\_\_x字段

3 

4

数据的使用注意什么时候该申请什么时候只该检查



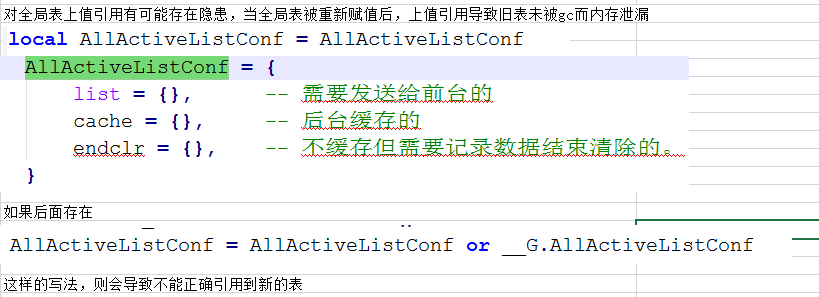
5

如果功能中不小心修改了配置表而不还原，从而导致对下一次操作的影响，可能造成严重的问题。（配置表只读化参见server/ZoneServer/script/main.lua function readonly() 请仔细阅读函数说明）

6

功能修改后提交测试，需要整理完整测试流程给测试，确保能够测到代码的每个部分

7



8

在函数接口中指定要操作的目标对象（部分老接口暂不支持）

