**Lua Code Style Guide -- Lua代码规范**

**Naming 命名规则11**常量和枚举值使用全大写用下划线\_分割单词

CAPS\_WITH\_UNDER

* 普通成员方法与变量, 本地变量与函数参数, 小写第一个单词

lowerFirstWords

* 单实例方法与变量全大写单词首字母

CapWords

* 只允许内部使用的方法与变量在名字前加下划线\_

\_BeginWithUnder

\_beginWithUnder

**Define and Usage 定义与使用**

**File and Package 文件和包**

* Define

文件名唯一, 全大写单词首字母

PackageName.lua

ClassName.lua

EnumName.lua

DataName.lua

* Usage
* local PackageName = require("PackageName")

**Enum 枚举类型**

EnumName =

{

ENUM\_VALUE\_NAME1 = 1,

ENUM\_VALUE\_NAME2 = 2,

}

**Function and Method 函数和成员方法**

* Function 函数
* Define
* local function PublicFunctionName(argName1, argName2, ...)
* -- TODO: Do something
* end
* local function \_InternalFunctionName(argName1, argName2, ...)
* -- TODO: Do something
* end
* Usage
* PublicFunctionName(argName1, argName2, ...)
* \_InternalFunctionName(argName1, argName2, ...)
* Method 成员方法
* Define
* function ClassName:publicFunctionName(argName1, argName2, ...)
* -- TODO: Do something
* end
* function ClassName:\_internalFunctionName(argName1, argName2, ...)
* -- TODO: Do something
* end
* Usage
* self:functionName(argName1, argName2, ...)
* Static Method 静态成员方法
* Define
* function ClassName.PublicFunctionName(argName1, argName2, ...)
* -- TODO: Do something
* end
* function ClassName.\_InternalFunctionName(argName1, argName2, ...)
* -- TODO: Do something
* end
* Usage
* ClassName.FunctionName(argName1, argName2, ...)

**Variable and Member 变量与成员变量**

* Variable 变量
* variableName = XXXXXX
* Const Variable 常量
* CONST\_VARIABLE\_NAME = XXXXXX
* Global Variable 全局变量
* GlobalVariableName = XXXXXX
* Member 成员变量
* Define
* self.publicVariableName = XXXXXX
* self.\_internalVariableName = XXXXXX
* Usage
* self.publicVariableName
* self.\_internalVariableName
* Static Member 静态成员变量
* Define
* ClassName.PublicVariableName = XXXXXX
* ClassName.\_InternalVariableName = XXXXXX
* Usage
* ClassName.PublicVariableName
* ClassName.\_InternalVariableName

**Comment 注释**

* Single-Line 单行注释
* -- Single Line Conment
* Multi-Line 多行注释
* --[[
* Multi-Line
* Conment
* ]]
* TODO 功能注释
* -- TODO: XXXXXX
* FIXME 提醒注释
* -- FIXME: XXXXXX

**Class 类**

**Sample Class SampleClass.lua**

-- TODO: require package

local Class = require("Class")

-- TODO: class define

local SampleClass = Class:extend -- class interitance

{

StaticVariable = nil, -- public static member variable

\_InternalStaticVariable = {}, -- internal static member variable

CONST\_VARIABLE = "Sample Const Variable", -- const variable

}

-- TODO: public static member function

function SampleClass.GetStaticVariable(arg, ...)

return SampleClass.\_GetStaticVariable(arg, ...)

end

-- TODO: internal static member function

function SampleClass.\_GetStaticVariable(arg, ...)

print(SampleClass.CONST\_VARIABLE)

if not SampleClass.StaticVariable == true then -- test StaticVariable is nil

return SampleClass.\_InternalStaticVariable

end

return nil

end

-- TODO: class constructor

function SampleClass:\_\_init(arg, ...)

self.publicMember = arg

self.\_internalMember = {}

end

-- TODO: public member function

function SampleClass:getMember(arg, ...)

return self:\_getMember(arg, ...)

end

-- TODO: internal member function

function SampleClass:\_getMember(arg, ...)

print(SampleClass.CONST\_VARIABLE) -- use const variable

if self.publicMember then -- test publicMember is not nil

return self.\_internalMember

end

return nil

end

return SampleClass

**Test Sample Class**

-- TODO: Sample Code

local SampleClass = require("SampleClass")

SampleClass.GetStaticVariable() -- call public static member function

local sample = SampleClass(0) -- new SampleClass

sample:getMember() -- call public member function

**使用规范**

**废弃module函数**

* 由于module函数有缺陷, 并且Lua后续的版本已经废弃module, 因此我们也不使用module
* 对module函数的讨论 [​http://lua-users.org/wiki/LuaModuleFunctionCritiqued](http://lua-users.org/wiki/LuaModuleFunctionCritiqued)

**使用require函数跨文件引用**

* require函数被重写, 不支持交叉引用

**Lua OOP 面向对象编程**

* [​http://lua-users.org/wiki/ObjectOrientedProgramming](http://lua-users.org/wiki/ObjectOrientedProgramming)