Atitit php编程开发规范总结

目录

[1. 一、规范前言篇 1](#_Toc12751)

[2. 命名 2](#_Toc24126)

[2.1. 见名知意 2](#_Toc32121)

[2.2. 单词间隔 2](#_Toc2977)

[3. 语法书写篇 2](#_Toc20813)

[4. 项目目录 2](#_Toc21709)

[4.1. 简单的三层结构 项目 模块 功能层次法 2](#_Toc7668)

[4.2. 结构可参照知名sdk层次法 2](#_Toc5112)

[5. 类库选型 2](#_Toc6198)

[6. 代码编写 3](#_Toc2185)

[6.1. 方法长度不要太长 超出 屏幕太多 3](#_Toc28822)

[6.2. 应优先使用dsl实现需要的功能 简化开发 3](#_Toc15569)

[6.3. 参数传递 如果参数太多可以使用数组组合 3](#_Toc31917)

[6.4. 优先单引号更加简单 3](#_Toc31328)

[6.5. 字符串变量拼接尽可能使用双引号法 3](#_Toc2212)

[6.6. Other。。。 3](#_Toc18990)

[7. 开发方法fp oop多范式支持 3](#_Toc31039)

[7.1. 保持Kiss简单原则，优先使用简单方法解决 3](#_Toc9973)

[7.2. 面向文件法 3](#_Toc27461)

[7.3. Fp 面向过程 函数式编程 4](#_Toc31597)

[7.4. Oop面向对象 4](#_Toc9053)

[7.5. 综合使用取长补短 4](#_Toc4933)

[8. 六、数据库应用篇 4](#_Toc31122)

[9. 源码版本管理git 4](#_Toc7442)

[10. 5 注释(Comments) 7 5](#_Toc1552)

[11. 其它说明篇 5](#_Toc3210)

[11.1. 避免复杂扩展 5](#_Toc16620)

[12. 代码范例(Code Examples) 5](#_Toc26566)

# 一、规范前言篇

# 命名

## 见名知意

如果一个名字不能见名知意，则需要重构。。一般使用英文，但非常见英文可读性会非常差，此时可以附加拼音。。

## 单词间隔

常用下划线或大小写间隔 作为一个整体

# 语法书写篇

不要嵌套过深

# 项目目录

## 简单的三层结构 项目 模块 功能层次法

## 结构可参照知名sdk层次法

Common cfg io net date str math等目录

# 类库选型

优先简单的类库，一般可以按照复杂度流行度，选出三个类库，简单，中等，复杂模式。。

从中选取

文档资料需要众多，小众类库不要选，除非只使用简单子集，并且确实更加简便让开发

# 代码编写

## 方法长度不要太长 超出 屏幕太多

一般一到俩屏幕为好。。

## 应优先使用dsl实现需要的功能 简化开发

Sql 正则表达式等 cli命令解释器模式 lamda表达式等

## 参数传递 如果参数太多可以使用数组组合

## 优先单引号更加简单

## 字符串变量拼接尽可能使用双引号法

1.给php变量赋值为字符串，尽量用单引号。单引号速度要快很多。

2.给php变量赋值时，值中带变量，就的用双引号了，双引号能自动解析变量，方便很多。

如$b=blue; $a="php$b"; echo $a;输出phpblue （单双引号各有千秋啊）

其次使用句号拼接法

## Other。。。

# 开发方法fp oop多范式支持

## 保持Kiss简单原则，优先使用简单方法解决

循序渐进，当简单方法不足以实现需求时，再使用更复杂方法

## 面向文件法

通过requie 文件来做代码复用调用 。。业务方法常用

## Fp 面向过程 函数式编程

通过函数来做代码复用调用，一般大多数使用其即可解决

## Oop面向对象

Oo设计，，代码更啰嗦些，但可解决复杂问题，当面对少数复杂问题使用oop解决

## 综合使用取长补短

# 六、数据库应用篇

数据库的设计尽可能符合三个范式（

数据库名称应该见名知意

以下划线分隔单词，避免跨平台时可能出现的大小写错误。

开发结束后，必须针对SQL查询语句的条件语句部分（where）添加索引，须匹配多个条件的应该使用聚合索引。

每当数据库（表）发生结构性变化时须登记保存；日常须定时（不超过三个工作日）备份数据库结构及其数据。

尽可能利用数据库的各项功能简化开发，适当使用，也不能滥用。。一般中小型项目更多的使用数据库各项功能。更大型项目只使用数据库的基本功能

# 源码版本管理git

# 5 注释(Comments) 7

5.1 实现注释的格式(Implementation Comment Formats) 8

5.1.1 块注释(Block Comments) 8

5.1.2 单行注释(Single-Line Comments) 9

5.1.3 尾端注释(Trailing Comments) 9

5.1.4 行末注释(End-Of-Line Comments) 10

5.2 文档注释(Documentation Comments) 10

# 其它说明篇

## 避免复杂扩展

可以使用其他语言类库来代替，通过cli互相调用

# 代码范例(Code Examples)