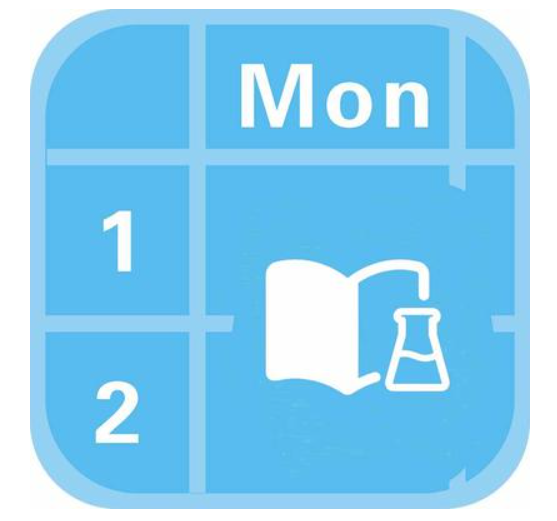
《课程备忘录》

代码规范文档



小 组： G10

小组成员 吴登钻31901236（组长）

赵晟浩 31901244 钟朱楠 31901245

**修 订 记 录**

**类别：**A – 增加 M – 修改 D – 删除

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **版本** | **日期** | **制定** | **类别** | **批准** | **描述（注明修改的条款或页）** |
| 0.1.0 | 2021-12-12 | 钟朱楠 | A | 吴登钻 | 确定文档目录结构，编写初稿 |
| 0.1.1 | 2021-12-14 | 钟朱楠 | A | 吴登钻 | 补充完善初稿，增加例子 |
| 0.2.0 | 2021-12-13 | 吴登钻 | M | 钟朱楠 | 修改部分细则 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

目录

[一、规范目的 2](#_Toc89632044)

[二、基本准则 2](#_Toc89632045)

[三、命名规范 2](#_Toc89632046)

[四、函数方法常用的动词 3](#_Toc89632048)

[五、编写注释 5](#_Toc89632049)

[六、编写注释 6](#_Toc89632050)

## 一、规范目的

为提高团队协作效率，便于前端后期优化维护，输出高质量的文档。

## 二、基本准则

**·**符合web标准，结构表现行为分离，兼容性优良。页面性能方面，代码要求简洁明了有序， 尽可能的减小服务器负载，保证最快的解析速度。

**·**项目的维护和二次开发可能是直接或间接的团队合作，所以创建易维护的代码是一个项目成功与否的关键，易维护的代码意味着具有如下特性：

1. 阅读性好：如良好的注释和命名规范，有文档
2. 具有一致性：看起来如同一个人编写
3. 代码的松耦合，高度模块化：将页面内的元素视为一个个模块，相互独立，尽量避免耦合过高的代码，从html,css,js三个层面都要考虑模块化
4. 严格按照规范编写代码

## 三、命名规范

1.代码中的命名均不能以下划线或美元符号开始，也不能以下划线或美元符号结束。 反例：\_name / \_\_name / $Object / name\_ / name$ / Object$

2.代码中的命名严禁使用拼音与英文混合的方式，更不允许直接使用中文的方式。说明：正确的英文拼写和语法可以让阅读者易于理解，避免歧义。注意，即使纯拼音命名方式也要避免采用。

正例：alibaba / taobao / youku / hangzhou 等国际通用的名称，可视同英文。

反例：DaZhePromotion [打折] / getPingfenByName() [评分] / int 某变量 = 3

3. 构造函数名使用 UpperCamelCase 风格，必须遵从驼峰形式，但以下情形例外：DO / BO / DTO / VO / AO

正例：MarcoPolo / UserDO / XmlService / TcpUdpDeal / TaPromotion

反例：macroPolo / UserDo / XMLService / TCPUDPDeal / TAPromotion

4.方法名、参数名、普通变量都统一使用 lowerCamelCase 风格，必须遵从驼峰形式。

正例： localValue / getHttpMessage() / inputUserId

5.常量命名全部大写，单词间用下划线隔开，力求语义表达完整清楚，不要嫌名字长。

正例：MAX\_STOCK\_COUNT

反例：MAX\_COUNT

6. **对象方法与事件响应函数**：对象方法命名使用对象类名+动词+名词形式；

例如： AddressGetEmail()，

7. **事件响应函数**：触发事件对象名+事件名或者模块名

例如：DivClick()，AddressSubmitButtonClick()

8. 例外情况

以根据项目及团队需要，设计出针对项目需要的前缀规范，从而达到团队开发协作便利的目的。

* 作用域不大临时变量可以简写，比如：str，num，bol，obj，fun，arr。
* 循环变量可以简写，比如：i，j，k等。

## 四、函数方法常用的动词

|  |  |  |  |
| --- | --- | --- | --- |
| get/获取 | set/设置 | add/增加 | remove/删除 |
| creat/创建 | destroy/移除 | start/启动 | stop/停止 |
| open/打开 | close/关闭 | read/读取 | write/写入 |
| load/载入 | save/保存 | begin/开始 | end/结束 |
| backup/备份 | restore/恢复 | import/导入 | export/导出 |
| split/分割 | merge/合并 | inject/注入 | extract/提取 |
| attach/附着 | detach/脱离 | bind/绑定 | separate/分离 |
| view/查看 | browse/浏览 | edit/编辑 | modify/修改 |
| select/选取 | mark/标记 | copy/复制 | paste/粘贴 |
| undo/撤销 | redo/重做 | insert/插入 | delete/移除 |
| add/加入 | append/添加 | clean/清理 | clear/清除 |
| index/索引 | sort/排序 | find/查找 | search/搜索 |
| increase/增加 | decrease/减少 | play/播放 | pause/暂停 |
| launch/启动 | run/运行 | compile/编译 | execute/执行 |
| debug/调试 | trace/跟踪 | observe/观察 | listen/监听 |
| build/构建 | publish/发布 | input/输入 | output/输出 |
| encode/编码 | decode/解码 | encrypt/加密 | decrypt/解密 |
| compress/压缩 | decompress/解压缩 | pack/打包 | unpack/解包 |
| parse/解析 | emit/生成 | connect/连接 | disconnect/断开 |
| send/发送 | receive/接收 | download/下载 | upload/上传 |
| refresh/刷新 | synchronize/同步 | update/更新 | revert/复原 |
| lock/锁定 | unlock/解锁 | submit/提交 | commit/交付 |
| push/推 | pull/拉 | expand/展开 | collapse/折叠 |
| enter/进入 | exit/退出 | abort/放弃 | quit/离开 |

## 五、编写注释

**·** 公共组件维护者和各栏目WD都需要在文件头部加上注释说明：

/\*\*

\*文件用途说明

\*作者姓名，联系方式

\*制作日期

\*\*/

**·** 大的模块注释方法：

//=========

//代码用途

//=========

**·** 小的模块注释方法：

//代码注释

注释单独一行，不要在代码后的同一行加注释

## 六、编写注释

**·** 在分开for循环的各个部分的分号之后：例如，for (var i = 0; i < 10; i +=1){…}

* 在for循环中初始化多个变量（i和最大值等）：for (var i = 0, max = 10; i < max; i += 1){…}
* 在限定数组项的逗号后面：var a = [1, 2, 3];
* 对象属性的逗号之后和将属性名和属性值分开的冒号之后：var o = {a: 1, b: 2};
* 分隔开函数中各个参数的逗号之后：myFunc(a, b, c)
* 在函数声明的大括号之前：function myFunc() {}
* 在匿名函数表达式之后：var myFunc = function () {};