# 注释

## **类**

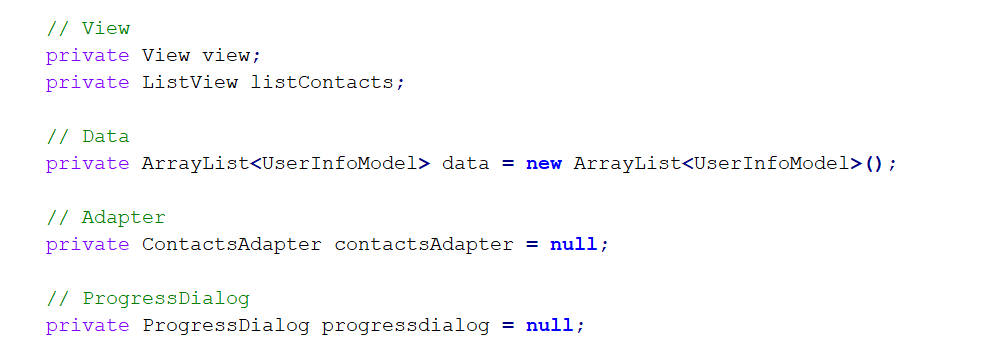
使用注释模板自动生成注释

|  |
| --- |
| 1. /\*\* 2. \* @Title: 标题 3. \* @Package 包名 4. \* @Description: 简述该类的作用 5. \* @author: 上次修改的作者 6. \* @date: 上次修改的时间和日期 7. \* @version 版本 8. \*/ 9. /\*\* 10. \* @Title: NetUtils 11. \* @Package com.qlnu.waterquery.net 12. \* @Description: 网络工具类，封装了网络请求 13. \* @author: Juston 14. \* @date: 2020-10-18 9:29 15. \* @version v1.2 16. \*/ |

实例成员

按照不同的功能区分开来 不同功能的实例成员中间用空行隔开并且用注释表明功能

实例成员应当放在类的前面 常量也应该放在前面



## 函数

使用注释模板自动生成注释 构造函数也是如此

|  |
| --- |
| 1. /\*\* 2. \* @Title: 标题 3. \* @Description: 简述该函数的作用 4. \* @param: 参数 5. \* @return: 返回值 6. \* @date: 上次修改的时间和日期 7. \* @throws 可能会抛出的异常 8. \*/ 9. /\*\* 10. \* @Title: getUserInfoById 11. \* @Description: 通过id获取用户信息 12. \* @param: id 唯一标识 13. \* @return: UserInfoModel 用户信息实体类 14. \* @date: 2020-10-1 12:32 15. \* @throws UserNotFoundException   \*/ |

## 语句



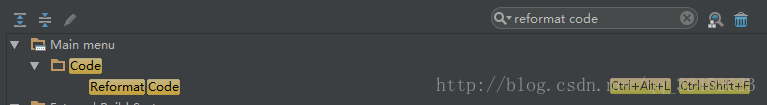
为了完成同一操作的代码应该放到一起 不同操作的代码之间应该用空行隔开

除非意图很明显的操作否则都应该加上一个简单的注释

如果有比较复杂的操作，应该详细解释代码是如何运行的

还有一点 就是缩进

使用系统自带的代码格式整理快捷键整理



# 命名规范

**命名尽可能的使用描述性的，少用意义不明的缩写**

|  |  |  |
| --- | --- | --- |
| **用处** | **用法** | **示例** |
| **类** | 大驼峰 | UpperCameCase |
| **常量** | 全部大写，下划线分开 | MAX\_SPEED |
| **函数** | 小驼峰 | getUserInfoById |
| **参数** | 小驼峰 | dateTime |
| **变量** | 小驼峰 私有变量前用m 静态变量前用s | mBtnLogin sPersonInfo |
| **资源id和资源文件** | 模块+功能名称 单词间下划线分开 都采用小写 | btn\_login |

mImgLogin 实例成员图片控件登录 要这样

缩写

|  |  |  |
| --- | --- | --- |
| 缩写 | 通用 | 意义 |
| btn | Button | 按钮 |
| img | Image | 图片 |
| str | String | 字符串 |
| init |  | 初始化 |
| bmp |  | 图片 |
| buf | Buffer | 缓冲区 |

待补充

## 另一些事情

不能直接在活动里面new线程做耗时操作（读写文件数据库、网络请求），而是应该交给service去做。

OnCreate只做一些简单的事情，不做耗时操作。

初始化应该交给onStart，然后在onStop释放。