**Android-Code-Style**

**1. 约定**

* **Activity.onCreate()，Fragment.onActivityCreated()，紧跟成员变量后，方法内部保持简单，尽量只调用initXXX()方法，如：initData()，initView()。**
* 调用方法保持**“临近原则”**，被调用的方法，放在调用方法下方
* **单个方法体不要过长**
* **代码任何地方不要拼错单词**
* 统一调整IDE的Tab缩进为4个空格
* **杜绝整个类代码格式化**

**2. 命名**

**2.1. 布局文件中的id命名**

**规则：使用驼峰命名，前缀+逻辑名称，类变量名和布局文件id名称保持一致，不需要下划线分割**

| **控件** | **缩写前缀** |
| --- | --- |
| TextView/EditText | text |
| ImageView | img |
| Button/RadioButton/ImageButton | btn |
| RelativeLayout/LinearLayout/FrameLayout | layout |
| ListView | listView |
| WebView | webView |
| CheckBox | checkBox |
| ProgressBar | progressBar |
| seekBar | seekBar |
| 其他控件 | 控件名首字母缩写作为前缀 |

* 如：TextView @+id/textTitle
* 如：EditView @+id/textName
* 如：Button @+id/btnSearch

**2.2. 布局文件命名**

**规则：** 使用**前缀\_逻辑名称**命名，单词全部小写，单词间以 下划线 分割。

| **布局类型** | **布局前缀** |
| --- | --- |
| Activity | activity\_ |
| Fragment | fragment\_ |
| Include | include\_ |
| Dialog | dialog\_ |
| PopupWindow | popup\_ |
| Menu | menu\_ |
| Adapter | layout\_item\_ |

**2.3. 资源文件命名**

**规则：** 使用 **前缀\_用途** 命名，单词全部小写，单词间以 下划线 分割。

* 图片资源文件命名

| **前缀** | **说明** |
| --- | --- |
| bg\_xxx | 各类背景图片 |
| btn\_xxx | 这种按钮没有其他状态 |
| ic\_xxx | 图标，一般用于单个图标 |
| bg\_描述\_状态1[\_状态2] | 用于控件上的不同状态 |
| btn\_描述\_状态1[\_状态2] | 用于按钮上的不同状态 |
| chx\_描述\_状态1[\_状态2] | 选择框，一般有2态和4态 |

* 第三方资源文件，不管在value、drawable

| **必须携带第三方资源前缀** |
| --- |
| umeng\_socialize\_style.xml |
| pull\_refresh\_attrs.xml |

**2.4. 类和接口命名**

**规则：** 使用驼峰规则，首字母必须大写，使用名词或名词词组。要求简单易懂，富于描述，不允许出现无意义或错误单词。

| **类** | **描述** | **例如** |
| --- | --- | --- |
| Application类 | Application为后缀标识 | XXXApplication |
| Activity类 | Activity为后缀标识 | 闪屏页面类SplashActivity |
| 解析类 | Handler为后缀标识 |  |
| 公共方法类 | Utils或Manager为后缀标识 |  |
| 线程池管理类 | ThreadPoolManager |  |
| 日志工具类 | LogUtils |  |
| 数据库类 | 以DBHelper后缀标识 | MySQLiteDBHelper |
| Service类 | 以Service为后缀标识 | 播放服务：PlayService |
| BroadcastReceiver类 | 以Broadcast为后缀标识 | 时间通知：TimeBroadcast |
| ContentProvider类 | 以Provider为后缀标识 | 单词内容提供者：DictProvider |
| 直接写的共享基础类 | 以Base为前缀 | BaseActivity,BaseFragment |

**2.5. 方法的命名**

**规则：** 使用驼峰规则，首字母必须小写，使用动词。要求简单易懂，富于描述，不允许出现无意义或错误单词。

| **方法** | **说明** |
| --- | --- |
| initXX() | 初始化相关方法，使用init为前缀标识，如初始化布局initView() |
| httpXX() | http业务请求方法，以http为前缀标识 |
| getXX() | 返回某个值的方法，使用get为前缀标识 |
| saveXX() | 与保存数据相关的，使用save为前缀标识 |
| deleteXX() | 删除操作 |
| resetXX() | 对数据重组的，使用reset前缀标识 |
| clearXX() | 清除数据相关的 |
| isXX() | 方法返回值为boolean型的请使用is或check为前缀标识 |
| processXX() | 对数据进行处理的方法，尽量使用process为前缀标识 |
| displayXX() | 弹出提示框和提示信息，使用display为前缀标识 |
| drawXXX() | 绘制数据或效果相关的，使用draw前缀标识 |

**2.6. 变量命名**

**规则：** 使用驼峰规则，首字母必须小写，使用名词或名词词组。要求简单易懂，富于描述，不允许出现无意义或错误单词。

* 成员变量命名，自定义变量前添加m前缀，布局控件变量不用添加m前缀
* 常量命名，全部大写，单词间用下划线隔开

**3. 其他规范**

* Activity继承BaseFragmentActivity或SwipeBackActivity，可以使用ButterKnife注解代替findViewById
* 方法
  + 拆分臃肿方法，每个方法只作一件事
  + 做同一个逻辑的方法，尽量靠近放到一块，方便查看
  + 不要使用 try catch 处理业务逻辑
  + 使用JSON工具类，不要手动解析和拼装数据
* 控制语句
  + 减少条件嵌套，不要超过3层
  + if判断使用“卫语句”，减少层级  
    if(obj != null) { doSomething(); }   
    修改为：   
    if(obj == null) { return; } doSomething();
  + if语句必须用{}包括起来,即便是只有一句
* 处理“魔数”等看不懂的神秘数字
  + 代码中不要出现数字，特别是一些标识不同类型的数字。
  + 所有意义数字全部抽取到Constant公共类中，避免散布在各位类中。
* 空行：空行将逻辑相关代码段隔开，简洁清楚，提高可读性
  + 成员变量之间，根据业务形成分组加空行
  + 方法之间加空行
* 用好TODO标记
  + 记录想法，记录功能点，开发过程中可以利用TODO记录一下临时想法或为了不打扰思路留下待完善的说明
  + 删除无用TODO，开发工具自动生成的TODO，或则已经完善的TODO，一定要删除。