# [Android 命名规范](http://www.cnblogs.com/zhangs1986/p/3776807.html)

在命名规范的统一下，在加上注释，日后维护会高效快捷好多

## 1 标识符命名法

1. 驼峰(Camel)命名法:又称小驼峰命名法，除首单词外，其余所有单词的第一个字母大写。
2. 下划线命名法:单词与单词间用下划线做间隔。
3. 帕斯卡(pascal)命名法:又称大驼峰命名法，所有单词的第一个字母大写
4. 匈牙利命名法:广泛应用于微软编程环境中，在以Pascal命名法的变量前附加小写序列说明该变量的类型。量的取名方式为：<scope\_> + <prefix\_> + <qualifier>范围前缀，类型前缀，限定词。

**个人觉得标识符命名原则**：**尽可能的用最容易识别的含义，推荐前两种**

## 2英文缩写原则

1. 较短的单词可通过去掉“元音”形成缩写
2. 较长的单词可取单词的头几个字母形成缩写
3. 此外还有一些约定成俗的英文单词缩写

**下面为常见的英文单词缩写**

|  |  |
| --- | --- |
| **名称** | **缩写** |
| icon | ic （主要用在app的图标） |
| color | cl（主要用于颜色值） |
| divider | di（主要用于分隔线，不仅包括Listview中的divider，还包括普通布局中的线） |
| selector | sl（主要用于某一view多种状态，不仅包括Listview中的selector，还包括按钮的selector） |
| average | avg |
| background | Bg（主要用于布局和子布局的背景） |
| buffer | buf |
| control | ctrl |
| delete | del |
| document | doc |
| error | err |
| escape | esc |
| increment | inc |
| infomation | info |
| initial | init |
| image | img |
| Internationalization | I18N |
| length | len |
| library | lib |
| message | msg |
| password | pwd |
| position | pos |
| server | srv |
| string | str |
| temp | tmp |
| window | wnd(win) |

## 3命名规范

### **包（packages）**

采用反域名命名规则，全部使用小写字母。一级包名为com，二级包名为xx（可以是公司或则个人的随便），三级包名根据应用进行命名，四级包名为模块名或层级名(com.tencent.qq.account)

|  |  |
| --- | --- |
| **包名** | **此包中包含** |
| com.xx.应用名称缩写.activity | 页面用到的Activity类 (activities层级名用户界面层) |
| com.xx.应用名称缩写.base | 页面中每个Activity类共用的类(如BaseActivity) |
| com.xx.应用名称缩写.adapter | 页面用到的Adapter类(适配器的类) |
| com.xx.应用名称缩写.util | 此包中包含：公共工具方法类（util模块名） |
| com.xx.应用名称缩写.bean(或则com.xx.应用名称缩写.model) | 此包中包含：元素类 |
| com.xx.应用名称缩写.db | 数据库操作类(sharepreference、sqlite、file) |
| com.xx.应用名称缩写.widget() | 自定义的View类等 |
| com.xx.应用名称缩写.service | Service服务 |
| com.xx.应用名称缩写.broadcast | Broadcast服务 |

### 类（classe）

名词，采用大驼峰命名法，尽量避免缩写，除非该缩写是众所周知的，比如HTML,URL，如果类名称中包含单词缩写，则单词缩写的每个字母均应写

|  |  |  |
| --- | --- | --- |
| **类** | **描述** | **例如** |
| activity类 | Activity为后缀标识 | 欢迎页面类WelcomeActivity |
| Adapter类 | Adapter 为后缀标识 | 新闻详情页面类 NewDetailAdapter |
| 解析类 | Hlr为后缀标识 | 首页解析类HomePosterHlr |
| 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：Logger |
| 数据库类 | 以DBHelper后缀标识 | 新闻数据库：NewDBHelper |
| Service类 | 以Service为后缀标识 | 时间服务TimeService |
| BroadcastReceive类 | 以Broadcast为后缀标识 | 时间通知TimeBroadcast |
| ContentProvider | 以Provider为后缀标识 | 联系人ContactProvider |
| 直接写的共享基础类 | 以Base开头 | BaseActivity,BaseFragment |

### 接口（interface）

命名规则与类一样采用大驼峰命名法，多以able或Listener结尾，如interface Runnable ;  
interface AccessListener 。

### 方法（methods）

动词或动名词，采用小驼峰命名法如:onCreate(),run()

|  |  |
| --- | --- |
| **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() |
| isXX() | checkXX()方法返回值为boolean型的请使用is或check为前缀标识 |
| getXX() | 返回某个值的方法，使用get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 |
| showXX() | 弹出提示框和提示信息，使用show为前缀标识 |
| saveXX() | 与保存数据相关的，使用sav为e前缀标识 |
| resetXX() | 对数据重组的，使用reset前缀标识 |
| clearXX() | 清除数据相关的 |
| removeXXX() | 清除数据相关的 |
| drawXXX() | 绘制数据或效果相关的，使用draw前缀标识 |

### 变量（variables）

采用小驼峰命名法。类中控件名称必须与xml布局id保持一致。

全局变量可使用m开头尽量不要超过三个单词组合，避免使用拼音，不会的可以百度翻译,太长可用缩写，如mTvAccountName;

### **常量（Constants）**

全部大写,采用下划线命名法，常量写在全局变量的最后，例如：MIN\_WIDTH

### 资源文件（图片drawable文件夹下）

全部小写，采用下划线命名法，加前缀区分:

|  |  |
| --- | --- |
| **名称** | **功能** |
| btn\_xx | 按钮图片使用btn\_整体效果**（selector）** |
| btn\_xx\_normal | 按钮图片使用btn\_正常情况效果 |
| btn\_xx\_press | 按钮图片使用btn\_点击时候效果 |
| bg\_head | 背景图片使用bg\_功能\_说明 |
| icon\_more\_help | 图标图片使用icon\_功能\_说明 |
| seg\_list\_line | 具有分隔特征的图片使用seg\_功能\_说明 |
| sel\_ok | 选择图标使用sel\_功能\_说明 |

命名后缀：

|  |  |
| --- | --- |
| **后缀** | **说明** |
| unit | 在使用xml的tilemode来配图片时，element图片使用此后缀 |
| nor | 图片的状态，代表普通状态 |
| hl | 图片的状态，代表高亮状态 |
| press | 图片的状态，代表按下状态 |
| select | 图片的状态，代表其所占的view被选中 |
| unselect | 图片的状态，代表其所占的view没有被选中 |

### 布局文件(xml)

Activity默认布局，以去掉后缀的Activity类进行命名。不加后缀：

#### ****功能模块命名****

例如：**activity\_main.xml、activity\_more.xml,fragment\_main**

#### Dialog命名

 例如：**dialog\_hint.xml**

#### PopupWindow命名

例如：**ppw \_info.xml**

#### 列表项命名

例如：**item\_city\_info.xml**

#### 包含项

例如：**include\_head.xml、include\_bottom.xml**

#### ****自定义控件命名****

**例如：widget\_group\_view.xml**

动画文件（anim文件夹下）

全部小写，采用下划线命名法，加前缀区分

|  |  |  |
| --- | --- | --- |
| **动画命名例子** | **规范写法** | **备注** |
| fade\_in | 淡入 |  |
| fade\_out | 淡出 |  |
| push\_down\_in | 从下方推入 |  |
| push\_down\_out | 从下方推出 |  |
| push\_left | 推像左方 |  |
| slide\_in\_from\_top | 从头部滑动进入 |  |
| zoom\_enter | 变形进入 |  |
| slide\_in | 滑动进入 |  |

### layout中的id命名

命名模式为：view缩写\_模块名称\_view的逻辑名称

|  |  |
| --- | --- |
| **控件** | **缩写** |
| LayoutView | lv |
| RelativeView | rv |
| TextView | tv |
| Button | btn |
| ImageButton | imgBtn |
| ImageView | iv |
| CheckBox | cb |
| RadioButton | rb |
| analogClock | anaClk |
| DigtalClock | dgtClk |
| DatePicker | dtPk |
| EditText | edt或et |
| TimePicker | tmPk |
| toggleButton | tglBtn |
| ProgressBar | pB |
| SeekBar | skBar |
| AutoCompleteTextView | autoTxt |
| ZoomControls | zmCtl |
| VideoView | vdoVi |
| WebView | wv |
| RantingBar | ratBar |
| Tab | tab |
| Spinner | spn |
| Chronometer | cmt |
| ScollView | sclVi |
| TextSwitch | txtSwt |
| ImageSwitch | imgSwt |
| listView | lv |
| ExpandableList | epdLt |
| MapView | mapVi |

## 4 Android编码规范建议

1. java代码中不出现中文，最多注释中可以出现中文
2. 局部变量命名、静态成员变量命名

只能包含字母，单词首字母除第一个外，都为大写，其他字母都为小写

1. 常量命名

只能包含字母和\_，字母全部大写，单词之间用\_隔开

1. 服务端可以实现的，就不要放在客户端
2. 引用第三方库要慎重，避免应用大容量的第三方库，导致客户端包非常大
3. 处理应用全局异常和错误，将错误以邮件的形式发送给服务端
4. 图片的.9处理
5. 使用静态变量方式实现界面间共享要慎重
6. Log(系统名称模块名称接口名称，详细描述)
7. 单元测试（逻辑测试、界面测试）
8. 不要重用父类的handler，对应一个类的handler也不应该让其子类用到，否则会导致message.what冲突
9. activity中在一个View.OnClickListener中处理所有的逻辑
10. strings.xml中使用%1$s实现字符串的通配
11. 如果多个Activity中包含共同的UI处理，那么可以提炼一个CommonActivity，把通用部分叫由它来处理，其他activity只要继承它
12. 如果所开发的为通用组件，为避免冲突，将drawable/layout/menu/values目录下的文件名增加前缀
13. 数据一定要效验，例如字符型转数字型，如果转换失败一定要有缺省值；服务端响应数据是否有效判断

## 5 Android结构建议

1. 布局采用RelativeLayout减少布局层级结构
2. 如果数据是服务器动态获取的，在布局里面不要写死
3. Activity里面的Log调试在发布的时候尽量去掉
4. Activity里面变量定义的分层可采用基本数据类型变量->view->adapter->对象->常量
5. Activity的跳转用static写在Activity内部onCreate()上面,

如public static void startGroupActivity(Context context,int orderId)

1. 尽量避免滚动控件的嵌套,可采用自定义View里面的addView的方式根据不同的数据量加载组合控件视图,页面的实现也可用自定义View拆分
2. 功能模块的实现时，考虑是否有可共用的控件，有则抽出，如倒计时、广告栏等等
3. 第三方库尽量抽成libarary复用，如网络库、图片库、刷新库、分享库等等
4. 个人代码风格要统一，如全部使用大小写或全部使用下划线，而不是各种混合，尽量不让Android studio提示命名问题
5. 统一标题栏、统一UI资源、统一页面跳转动画、统一空视图无网络页面、统一Dialog、统一Toast、统一Loading，这里的统一是指抽出来封装公用。