|  |
| --- |
| 北京泰合佳通信息技术有限公司 |
| Android开发手册 |
| CMMI 3体系文档 |

目录

[1 前言 4](#_Toc509912950)

[1.1 目的 4](#_Toc509912951)

[1.2 适用范围 4](#_Toc509912952)

[1.3 规则级别定义 4](#_Toc509912953)

[2 工程规范 4](#_Toc509912954)

[2.1 开发环境 4](#_Toc509912955)

[2.2 目录结构 5](#_Toc509912956)

[2.3 提交规范 6](#_Toc509912957)

[3 排版规范 7](#_Toc509912958)

[3.1 代码风格 7](#_Toc509912959)

[4 命名规范 7](#_Toc509912960)

[4.1 包命名 7](#_Toc509912961)

[4.2 类和接口命名 7](#_Toc509912962)

[5 注释规范 8](#_Toc509912963)

[6 资源文件规范 8](#_Toc509912964)

[6.1 【强制】动画资源文件（anim/ 和 animator/） 8](#_Toc509912965)

[6.2 【强制】颜色资源文件（color/） 9](#_Toc509912966)

[6.3 【强制】图片资源文件（drawable/ 和 mipmap/） 9](#_Toc509912967)

[6.4 【强制】布局资源文件（layout/） 11](#_Toc509912968)

[6.5 【强制】菜单资源文件（menu/） 11](#_Toc509912969)

[6.6 【强制】values 资源文件（values/） 11](#_Toc509912970)

[6.6.1 【强制】colors.xml 12](#_Toc509912971)

[6.6.2 dimens.xml 13](#_Toc509912972)

[6.6.3 strings.xml 14](#_Toc509912973)

[6.6.4 styles.xml 14](#_Toc509912974)

[6.6.5 id 命名 15](#_Toc509912975)

[7 测试规范 15](#_Toc509912976)

[7.1 【建议】单元测试 15](#_Toc509912977)

[7.2 【建议】Espresso 测试 16](#_Toc509912978)

[8 编程规范 16](#_Toc509912979)

[9 参考文献 16](#_Toc509912980)

**修订记录**

|  |  |  |  |
| --- | --- | --- | --- |
| 时间 | 版本号 | 修订内容 | 修订人 |
| 2018年3月22日 | **1.0** | **起草** | **闫季群** |
| **2018/5/22** | **1.0.1** | **更新开发环境** | **闫季群** |
|  |  |  |  |
|  |  |  |  |

**审核记录**

|  |  |  |  |
| --- | --- | --- | --- |
| 时间 | 版本号 | 审核结果 | 审核人 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 前言

## 目的

为了有利于项目维护、增强代码可读性、提升 Code Review 效率以及规范团队安卓开发，故提出以下安卓开发规范。

## 适用范围

本规范适用于北京泰合佳通信息技术有限公司所有使用Java作为开发语言的Android软件产品。所有使用Java作为开发语言的Android软件产品都须按照本规范的内容进行编码。

## 规则级别定义

|  |  |
| --- | --- |
| **级别** | **定义** |
| 强制 | 编程时必须遵守的规则 |
| 建议 | 编程时必须加以考虑的规则 |

# 工程规范

## 开发环境

* 【强制】使用统一的开发环境进行开发，对以下开发工具进行版本约束：
* Android Studio环境要求

|  |  |
| --- | --- |
| **开发工具** | **版本** |
| jdk | 8u111 |
| AndroidStudio | 3.0.1 |
| Gradle | 4.1 |
| Gradle plugin | 3.0.1 |
|  |  |

Eclips环境要求

|  |  |
| --- | --- |
| **开发工具** | **版本** |
| jdk | 8u111 |
| Eclips | Oxygen |
| Android SDK Tools | 27 |
|  |  |

* 【强制】源文件编码方式采用UTF-8字符集。

## 目录结构

* 【建议】使用统一的工程目录结构，可选择如下两种：
* MVC开发方式的包结构定义

|  |  |
| --- | --- |
| **相对工程根目录的位置** | **描述** |
| com/domain/app | 基础包名，模版目录 |
| com/domain/app/App.java | 定义Application类 |
| com/domain/app/Config.java | 定义配置数据 |
| com/domain/app/base | 基础组件目录 |
| com/domain/app/custom\_view | 自定义视图目录 |
| com/domain/app/data | 数据处理 |
| com/domain/app/data/DataManager.java | 数据管理器 |
| com/domain/app/data/local | 来源于本地的数据，比如 SP，Database，Fil |
| com/domain/app/data/model | 定义 model（数据结构以及 getter/setter、compareTo、equals 等等，不含复杂操作） |
| com/domain/app/data/remote | 来源于远端的数据 |
| com/domain/app/feature | 功能 |
| com/domain/app/feature/feature1 | 功能1包含feature0Activity.java、feature0Fragment.java、xxAdapter.java、其他 class |
| com/domain/app/injection | 依赖注入实现类 |
| com/domain/app/util | util 工具类 |
| com/domain/app/widget | widget 小部件 |
|  |  |
|  |  |
|  |  |

* MVP开发方式的包结构定义

|  |  |
| --- | --- |
| 相对工程根目录的位置 | 描述 |
| com/domain/app | 基础包名，模版目录 |
| com/domain/app/App.application | Application文件 |
| com/domain/app/receiver | 接收者文件目录 |
| com/domain/app/service | 服务文件目录 |
| com/domain/app/di | 依赖注入对象 |
| com/domain/app/di/compoment | Dagger需要的compoment |
| com/domain/app/di/module | Dagger需要的module |
| com/domain/app/mvp | 存放Mvp对象 |
| com/domain/app/mvp/feature1 | 功能1 |
| com/domain/app/mvp/feature1/contract | 存放契约类 |
| com/domain/app/mvp/feature1/model | 存放数据访问类 |
| com/domain/app/mvp/feature1/model/api | 存放网络请求类 |
| com/domain/app/mvp/feature1/model/entity | 实体类 |
| com/domain/app/mvp/feature1/presenter | 存放presenter |
| com/domain/app/mvp/feature1/ui | UI层 |
| com/domain/app/util | util 工具类 |
| com/domain/app/widget | widget 小部件 |
|  |  |

## 提交规范

* 依照java编码规范

# 排版规范

## 代码风格

代码风格使用AndroidStudio推荐风格。在提交代码前进行代码排版格式化。

# 命名规范

## 包命名

* 依照java编码规范

## 类和接口命名

1. 依照java编码规范。
2. 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大写。

|  |  |  |
| --- | --- | --- |
| **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 | 新闻详情适配器 NewsDetailAdapter |
| **解析类** | Parser 为后缀标识 | 首页解析类 HomePosterParser |
| **工具方法类** | Utils 或 Manager 为后缀标识 | 线程池管理类：ThreadPoolManager 日志工具类：LogUtils（Logger 也可） 打印工具类：PrinterUtils |
| **数据库类** | 以 DBHelper 后缀标识 | 新闻数据库：NewsDBHelper |
| **Service 类** | 以 Service 为后缀标识 | 时间服务 TimeService |
| **BroadcastReceiver 类** | 以 Receiver 为后缀标识 | 推送接收 JPushReceiver |
| **ContentProvider 类** | 以 Provider 为后缀标识 | ShareProvider |
| **自定义的共享基础类** | 以 Base 开头 | BaseActivity, BaseFragment |

# 注释规范

* 依照java编码规范
* 环境配置建议使用JavaDoc2 plugin。
  + Preferences-》plugins-》search-》javadoc2
  + 如果没有，点击search in respositories，安装javaDoc2 ，安装后重启AS

# 资源文件规范

资源文件命名为全部小写，采用下划线命名法。

## 【强制】动画资源文件（anim/ 和 animator/）

安卓主要包含属性动画和视图动画，其视图动画包括补间动画和逐帧动画。属性动画文件需要放在 res/animator/ 目录下，视图动画文件需放在 res/anim/ 目录下。

命名规则：{模块名\_}逻辑名称。

说明：{} 中的内容为可选，逻辑名称 可由多个单词加下划线组成。

例如：refresh\_progress.xml、market\_cart\_add.xml、market\_cart\_remove.xml。

如果是普通的补间动画或者属性动画，可采用：动画类型\_方向 的命名方式。

例如：

| **名称** | **说明** |
| --- | --- |
| 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 | 推向左方 |
| slide\_in\_from\_top | 从头部滑动进入 |
| zoom\_enter | 变形进入 |
| slide\_in | 滑动进入 |
| shrink\_to\_middle | 中间缩小 |

## 【强制】颜色资源文件（color/）

专门存放颜色相关的资源文件。

命名规则：类型\_逻辑名称。

例如：sel\_btn\_font.xml。

颜色资源也可以放于 res/drawable/ 目录，引用时则用 @drawable 来引用，但不推荐这么做，最好还是把两者分开。

## 【强制】图片资源文件（drawable/ 和 mipmap/）

res/drawable/ 目录下放的是位图文件（.png、.9.png、.jpg、.gif）或编译为可绘制对象资源子类型的 XML 文件，而 res/mipmap/ 目录下放的是不同密度的启动图标，所以 res/mipmap/ 只用于存放启动图标，其余图片资源文件都应该放到 res/drawable/ 目录下。

命名规则：类型{\_模块名}\_逻辑名称、类型{\_模块名}\_颜色。

说明：{} 中的内容为可选；类型 可以是[可绘制对象资源类型](https://link.jianshu.com?t=%E5%8F%AF%E7%BB%98%E5%88%B6%E5%AF%B9%E8%B1%A1%E8%B5%84%E6%BA%90%E7%B1%BB%E5%9E%8B" \t "_blank)，也可以是控件类型（具体见附录[UI 控件缩写表](#ui-%E6%8E%A7%E4%BB%B6%E7%BC%A9%E5%86%99" \t "_blank)）；最后可加后缀 \_small 表示小图，\_big 表示大图。

例如：

| **名称** | **说明** |
| --- | --- |
| btn\_main\_about.png | 主页关于按键 类型\_模块名\_逻辑名称 |
| btn\_back.png | 返回按键 类型\_逻辑名称 |
| divider\_maket\_white.png | 商城白色分割线 类型\_模块名\_颜色 |
| ic\_edit.png | 编辑图标 类型\_逻辑名称 |
| bg\_main.png | 主页背景 类型\_逻辑名称 |
| btn\_red.png | 红色按键 类型\_颜色 |
| btn\_red\_big.png | 红色大按键 类型\_颜色 |
| ic\_head\_small.png | 小头像图标 类型\_逻辑名称 |
| bg\_input.png | 输入框背景 类型\_逻辑名称 |
| divider\_white.png | 白色分割线 类型\_颜色 |
| bg\_main\_head.png | 主页头部背景 类型\_模块名\_逻辑名称 |
| def\_search\_cell.png | 搜索页面默认单元图片 类型\_模块名\_逻辑名称 |
| ic\_more\_help.png | 更多帮助图标 类型\_逻辑名称 |
| divider\_list\_line.png | 列表分割线 类型\_逻辑名称 |
| sel\_search\_ok.xml | 搜索界面确认选择器 类型\_模块名\_逻辑名称 |
| shape\_music\_ring.xml | 音乐界面环形形状 类型\_模块名\_逻辑名称 |

如果有多种形态，如按钮选择器：sel\_btn\_xx.xml，采用如下命名：

| **名称** | **说明** |
| --- | --- |
| sel\_btn\_xx | 作用在 btn\_xx 上的 selector |
| btn\_xx\_normal | 默认状态效果 |
| btn\_xx\_pressed | state\_pressed 点击效果 |
| btn\_xx\_focused | state\_focused 聚焦效果 |
| btn\_xx\_disabled | state\_enabled 不可用效果 |
| btn\_xx\_checked | state\_checked 选中效果 |
| btn\_xx\_selected | state\_selected 选中效果 |
| btn\_xx\_hovered | state\_hovered 悬停效果 |
| btn\_xx\_checkable | state\_checkable 可选效果 |
| btn\_xx\_activated | state\_activated 激活效果 |
| btn\_xx\_window\_focused | state\_window\_focused 窗口聚焦效果 |

注意：使用 Android Studio 的插件 SelectorChapek 可以快速生成 selector，前提是命名要规范。

## 【强制】布局资源文件（layout/）

命名规则：类型\_模块名、类型{\_模块名}\_逻辑名称。

说明：{} 中的内容为可选。

例如：

| **名称** | **说明** |
| --- | --- |
| activity\_main.xml | 主窗体 类型\_模块名 |
| activity\_main\_head.xml | 主窗体头部 类型\_模块名\_逻辑名称 |
| fragment\_music.xml | 音乐片段 类型\_模块名 |
| fragment\_music\_player.xml | 音乐片段的播放器 类型\_模块名\_逻辑名称 |
| dialog\_loading.xml | 加载对话框 类型\_逻辑名称 |
| ppw\_info.xml | 信息弹窗（PopupWindow） 类型\_逻辑名称 |
| item\_main\_song.xml | 主页歌曲列表项 类型\_模块名\_逻辑名称 |

## 【强制】菜单资源文件（menu/）

菜单相关的资源文件应放在该目录下。

命名规则：{模块名\_}逻辑名称

说明：{} 中的内容为可选。

例如：main\_drawer.xml、navigation.xml。

## 【强制】values 资源文件（values/）

values/ 资源文件下的文件都以 s 结尾，如 attrs.xml、colors.xml、dimens.xml，起作用的不是文件名称，而是 <resources> 标签下的各种标签，比如 <style> 决定样式，<color> 决定颜色，所以，可以把一个大的 xml 文件分割成多个小的文件，比如可以有多个 style 文件，如 styles.xml、styles\_home.xml、styles\_item\_details.xml、styles\_forms.xml。

### colors.xml

<color> 的 name 命名使用下划线命名法，在你的 colors.xml 文件中应该只是映射颜色的名称一个 ARGB 值，而没有其它的。不要使用它为不同的按钮来定义 ARGB 值。

例如，不要像下面这样做：

<resources>

<color name="button\_foreground">#FFFFFF</color>

<color name="button\_background">#2A91BD</color>

<color name="comment\_background\_inactive">#5F5F5F</color>

<color name="comment\_background\_active">#939393</color>

<color name="comment\_foreground">#FFFFFF</color>

<color name="comment\_foreground\_important">#FF9D2F</color>

...

<color name="comment\_shadow">#323232</color>

使用这种格式，会非常容易重复定义 ARGB 值，而且如果应用要改变基色的话会非常困难。同时，这些定义是跟一些环境关联起来的，如 button 或者 comment，应该放到一个按钮风格中，而不是在 colors.xml 文件中。

相反，应该这样做：

<resources>

<!-- grayscale -->

<color name="white" >#FFFFFF</color>

<color name="gray\_light">#DBDBDB</color>

<color name="gray" >#939393</color>

<color name="gray\_dark" >#5F5F5F</color>

<color name="black" >#323232</color>

<!-- basic colors -->

<color name="green">#27D34D</color>

<color name="blue">#2A91BD</color>

<color name="orange">#FF9D2F</color>

<color name="red">#FF432F</color>

</resources>

向应用设计者那里要这个调色板，名称不需要跟 "green"、"blue" 等等相同。"brand\_primary"、"brand\_secondary"、"brand\_negative" 这样的名字也是完全可以接受的。像这样规范的颜色很容易修改或重构，会使应用一共使用了多少种不同的颜色变得非常清晰。通常一个具有审美价值的 UI 来说，减少使用颜色的种类是非常重要的。

注意：如果某些颜色和主题有关，那就单独写一个 colors\_theme.xml。

### dimens.xml

像对待 colors.xml 一样对待 dimens.xml 文件，与定义颜色调色板一样，你同时也应该定义一个空隙间隔和字体大小的“调色板”。 一个好的例子，如下所示：

<resources>

<!-- font sizes -->

<dimen name="font\_22">22sp</dimen>

<dimen name="font\_18">18sp</dimen>

<dimen name="font\_15">15sp</dimen>

<dimen name="font\_12">12sp</dimen>

<!-- typical spacing between two views -->

<dimen name="spacing\_40">40dp</dimen>

<dimen name="spacing\_24">24dp</dimen>

<dimen name="spacing\_14">14dp</dimen>

<dimen name="spacing\_10">10dp</dimen>

<dimen name="spacing\_4">4dp</dimen>

<!-- typical sizes of views -->

<dimen name="button\_height\_60">60dp</dimen>

<dimen name="button\_height\_40">40dp</dimen>

<dimen name="button\_height\_32">32dp</dimen>

</resources>

布局时在写 margins 和 paddings 时，你应该使用 spacing\_xx 尺寸格式来布局，而不是像对待 string 字符串一样直接写值，像这样规范的尺寸很容易修改或重构，会使应用所有用到的尺寸一目了然。 这样写会非常有感觉，会使组织和改变风格或布局非常容易。

### strings.xml

<string> 的 name 命名使用下划线命名法，采用以下规则：{模块名\_}逻辑名称，这样方便同一个界面的所有 string 都放到一起，方便查找。

| **名称** | **说明** |
| --- | --- |
| main\_menu\_about | 主菜单按键文字 |
| friend\_title | 好友模块标题栏 |
| friend\_dialog\_del | 好友删除提示 |
| login\_check\_email | 登录验证 |
| dialog\_title | 弹出框标题 |
| button\_ok | 确认键 |
| loading | 加载文字 |

### styles.xml

<style> 的 name 命名使用大驼峰命名法，几乎每个项目都需要适当的使用 styles.xml 文件，因为对于一个视图来说，有一个重复的外观是很常见的，将所有的外观细节属性（colors、padding、font）放在 styles.xml 文件中。 在应用中对于大多数文本内容，最起码你应该有一个通用的 styles.xml 文件，例如：

<style name="ContentText">

<item name="android:textSize">@dimen/font\_normal</item>

<item name="android:textColor">@color/basic\_black</item>

</style>

应用到 TextView 中：

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/price"

style="@style/ContentText"

/>

或许你需要为按钮控件做同样的事情，不要停止在那里，将一组相关的和重复 android:xxxx 的属性放到一个通用的 <style> 中。

### id 命名

命名规则：view缩写{\_模块名}\_逻辑名，例如： btn\_main\_search、btn\_back。

如果在项目中有用黄油刀的话，使用 AS 的插件：ButterKnife Zelezny，可以非常方便帮助你生成注解；没用黄油刀的话可以使用 Android Code Generator 插件。

# 测试规范

业务开发完成之后，开发人员做单元测试，单元测试完成之后，保证单元测试全部通过，同时单元测试代码覆盖率达到一定程度（这个需要开发和测试约定，理论上越高越好），开发提测。

## 【建议】单元测试

测试类的名称应该是所测试类的名称加 Test，我们创建 DatabaseHelper 的测试类，其名应该叫 DatabaseHelperTest。

测试函数被 @Test 所注解，函数名通常以被测试的方法为前缀，然后跟随是前提条件和预期的结果。

* 模板：void methodName前提条件和预期结果()
* 例子：void signInWithEmptyEmailFails()

注意：如果函数足够清晰，那么前提条件和预期的结果是可以省略的。

有时一个类可能包含大量的方法，同时需要对每个方法进行几次测试。在这种情况下，建议将测试类分成多个类。例如，如果 DataManager 包含很多方法，我们可能要把它分成 DataManagerSignInTest、DataManagerLoadUsersTest 等等。

## 【建议】Espresso 测试

每个 Espresso 测试通常是针对 Activity，所以其测试名就是其被测的 Activity 的名称加 Test，比如 SignInActivityTest。

# 编程规范

参考阿里巴巴android开发手册

# 参考文献

[Google Java 编程风格规范](https://link.jianshu.com/?t=http://www.hawstein.com/posts/google-java-style.html)

[Google 官方 Android 编码风格规范](https://link.jianshu.com/?t=http://www.360sdn.com/2013/android_0517/144.html)

谷歌相应的 Sample：[**todo-mvp**](https://link.jianshu.com/?t=https://github.com/googlesamples/android-architecture/tree/todo-mvp/)