Thursday, September 27, 2018

砸立 Android 开发手册

新同事必看

William yangchen_0808@163.com
上海砸立电子商务有限公司

版本更新说明

版本号	制定人	更新日期	备注
1.0.0	杨陈	2018-9-27	完成第一版砸立安卓开发手册，持续更新中

Android 新人必读

一、代码规范

1. 代码尽量简洁规范，不要做多余的 for 循环；
2. 代码尽量不要有报黄颜色的警告标识，编译器报的警告要引起重视并解决；
3. 关于 lambda 表达式和 jdk 高版本 API 的使用说明，相关 API 操作可能不支持，因为低系统版本手机会不支持。

   不支持的：List.stream().forEach();

   支持的：View.setOnClickListener(v-> {// TODO });

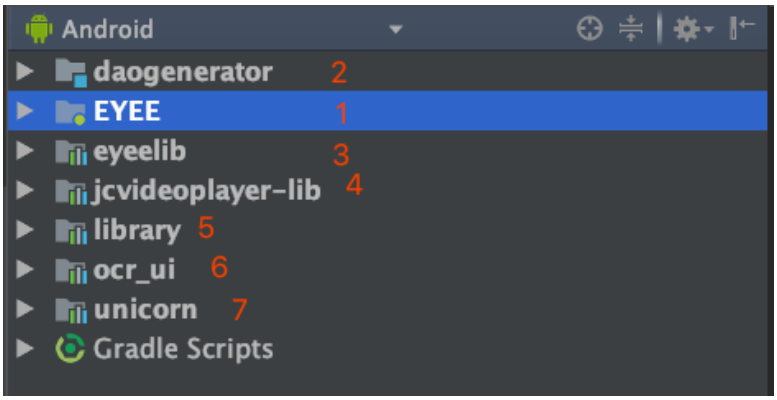
   View.setOnClickListener((v)-> {// TODO });

   View.setOnClickListener((View v)->{// TODO});// 多个参数同理

   View.setOnClickListener(this::methodName);
4. 为了提高代码可读性，提交代码前要格式化代码。重要的逻辑处需要将上注释，避免注释长篇大论。
5. 避免创建过多的临时变量，需要时才创建；避免类变量声明时就实例化，以免内存出现无法释放的情况。
6. 对于可能会出现 NPE 的代码处要加上非空判断，减少 crash。
7. 接口返回的数据模型 javaBean 要实现 Serializable，其内部引用的类型也需要实现 Serializable。模型中尽量不要出现定义接口引用 Activity 实例的行为。JavaBean 只做数据解析和临时存储。持久化的存储要用 SharedPreferences 和 SQLite 存储。
8. 定义的变量名要有意义，简短明了。数据模型定义一般是以 Bean 结尾；接口以字母 I 开头；
   a) Activity 命名以 Activity 结尾；
   b) Fragment，dialog 同理；
   c) 布局文件以 activity_xxx，dialog_xxx，fragment_xxx 等开头；
   d) drawable 文件以 shape_xxx, selector_xxx, layer_xxx, gradient_xxx 等开头；
   e) 点 9 图片放在 drawable 目录下，2 倍图放在 drawable-xhdpi，3 倍图放在 drawable-xxhdpi。目前项目适配 2 套图（2 倍和 3 倍）。设计稿的 1 倍图宽为 375px。

二、项目结构说明

1. module 结构



1 处：主项目；

2 处：GreenDAO java 工程；

3 处：项目基础依赖库，存放了大部分的基础组件和工具类；

4 处：jc 视频播放器依赖库；

5 处：下拉刷新和加载更多的依赖库；

6 处：百度 OCR 身份证识别依赖库

7 处：网易七鱼客服系统依赖库。

2. 主项目包级结构

1：基类

2：数据库 bean 和工具类

3：MVP 实现相关类

4：网络请求及配置

5：adapter 类

6：B2C 售后相关类

7：B2C 相关主页面

8：店铺相关类

9：C2C 售后相关类

10：优惠券相关类

11：5 个主页面

12：拼团相关类

13：签到兑换相关类

14：订单相关类

15：支付帮助类

16：商品相关类

17：球鞋 C2C 相关类

18：购物车相关类

19：资讯相关类

20：用户相关类

21：分享框实现

22：第三方登录

23：闪屏页

24：主页

25：工具包

26：自定义 View

27：H5 交互相关

28：基础配置相关

29：上下文

30：支付列表

3. gradle 依赖配置

新配置	弃用配置	行为
`implementation`	`compile`	Gradle将依赖项添加到编译类路径，并将依赖项打包到构建输出。但是，当您的模块配置`implementation`依赖项时，它让Gradle知道您不希望模块在编译时将依赖项泄漏给其他模块。也就是说，依赖性仅在运行时可用于其他模块。 使用这种依赖性配置，而不是`api`或`compile`（不建议使用）可导致**显著编译时间的改进**，因为它减少了构建系统需要重新编译的模块的数量。例如，如果`implementation`依赖项更改其API，则Gradle仅重新编译该依赖项以及直接依赖于它的模块。大多数应用和测试模块都应使用此配置。
`api`	`compile`	Gradle将依赖项添加到编译类路径并构建输出。当一个模块包含一个`api`依赖项时，它让Gradle知道该模块想要将该依赖项传递给其他模块，以便它们在运行时和编译时都可用。 此配置的行为就像`compile`（现在已弃用），但您应谨慎使用它，并且只能将您需要的依赖项可传递地导出到其他上游使用者。这是因为，如果`api`依赖项更改其外部API，Gradle将重新编译在编译时有权访问该依赖项的所有模块。因此，拥有大量`api`依赖项可以显着增加构建时间。除非您希望将依赖项的API公开给单独的模块，否则库模块应该使用`implementation`依赖项。
`compileOnly`	`provided`	Gradle仅将依赖项添加到编译类路径（即，它不会添加到构建输出中）。这在您创建Android模块时非常有用，并且在编译期间需要依赖项，但在运行时将其存在是可选的。 如果使用此配置，则库模块必须包含运行时条件以检查依赖项是否可用，然后正常更改其行为，以便在未提供时仍可正常运行。这有助于通过不添加不重要的瞬态依赖项来减小最终APK的大小。此配置的行为就像`provided`（现在已弃用）。
`runtimeOnly`	`apk`	Gradle仅将依赖项添加到构建输出，以便在运行时使用。也就是说，它不会添加到编译类路径中。此配置的行为就像`apk`（现在已弃用）。
`annotationProcessor`	`compile`	要添加对作为注释处理器的库的依赖关系，必须使用`annotationProcessor`配置将其添加到注释处理器类路径 。这是因为使用此配置可以通过将编译类路径与注释处理器类路径分开来提高构建性能。如果Gradle在编译类路径上找到注释处理器，它将停用 编译避免，这会对构建时间产生负面影响（Gradle 5.0及更高版本忽略编译类路径上的注释处理器）。 Android Gradle Plugin假定依赖是一个注释处理器，如果它的JAR文件包含以下文件：。如果插件检测到编译类路径上的注释处理器，则会产生构建错误。 `META-INF/services/javax.annotation.processing.Processor`

4. android studio 配置
   a) 截止到当前更新，android studio 的使用的版本为 Android Studio 3.1.4
      i. Build #AI-173.4907809, built on July 24, 2018
      ii. JRE: 1.8.0_152-release-1024-b01 x86_64
      iii. JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o
      iv. Mac OS X 10.14
   b) Gradle 的版本为 4.4
      i. 相关 gradle 命令及配置使用请参考 gradle 官网和谷歌官网学习。

5. git 使用
   a) 学习网址：git 工具学习使用
   b) 分支管理
      i. 目前项目有 master 分支，如果有新功能，会在 master 基础上新建分支，例如新需求红包功能 redpacket 分支（以下简称 rp）。
      ii. 各开发同事有对应远程 rp 的本地分支 l_rp，然后以 l_rp 为基础新建自己本地分支 local。各同事在自己的 local 分支上开发后，请参照以下步骤操作：
         1. 将 local 提交到本地仓库 。命令如下：
            git add .
            git commit –m “action desc”
         2. 切换到 l_rp 分支，然后拉取最新代码；
            git checkout l_rp
            git pull
         3. 切换到 local 分支，然后将 l_rp 合并到 local。(如果已知远程 rp 无最新提交，第 2 步可省略。直接将 local 合并到 l_rp)
            git checkout local
            git merge –no-ff –m “merge action desc” l_rp
            如果有代码冲突，解决冲突。无法解决时找冲突相关同事一起解决。
            git checkout l_rp
            git merge –no-ff –m “merge action desc” local
            git push
            如果已知远程 rp 无最新提交，则依次执行以下操作
            git merge –no-ff –m “merge action desc” local
            git push
         4. 第 3 步完成后，切换回 local 分支
            git checkout local
      iii. 至此已完成一次提交推送操作。至于其他相关命令操作请不熟悉的同事结合自己的 remote github repo 多加练习。
      iv. 另外附一份 git 常用命令的文档。详情请见《git 常用命令》。