## 2023年夏季Java小学期大作业

## 实验报告

王浩然 计23 2022010229

1. **代码结构**

项目位于app文件夹中，其主要子目录为java/com/java/wanghaoran; res，另外有manifest（应用Activity设置、网络权限申请等），及app级build.gradle（gradle配置文件，需要声明需要使用的第三方库等），下面分别主要说明。

1. java/com/java/wanghaoran 项目java文件库
   1. containers/ 储存项目所用容器

Keywords.java 存储九个关键词，方便日后添加分类，避免硬编码

News.java 存储新闻相关信息

NewsResponse.java 存储获得的请求

User.java 与SQLite配合，存储登录信息

* 1. service/ 存储项目相关服务

APIManager.java 发送、处理API请求

DBManager.java 处理用户独立数据库

NewsManager.java 控制Recommend和Search页的NewsList

HttpManager.java 控制APIManager和NewsManager间交互

MySQLiteOpenHelper.java 控制新闻数据库

UserSQLiteOpenHelper.java 控制用户数据库

* 1. ui/ 存储ui布局对应相应

newslist/{NewslistFragment.java, NewslistViewModel.java}

profile/{ProfileFragment.java, ProfileViewModel.java}

recomment/{RecommentFragment.java, RecommentViewModel.java}

RecyclerViewScrollListener.java 新闻列表展示控件

NewsDetailFragment.java 新闻具体内容展示控件

NewsListAdapter.java 新闻列表控件

NewsListFragment.java 新闻列表展示控件

RecordListFragment.java history/favorite页控件

SearchFragment.java search页控件

SelectPaddleAdapter.java recommend页左上角按钮选中后paddle控件

SelectPaddleFragment.java recommend页左上角按钮选中后paddle控件

TabListFragment.java recommend页上部tab控件

UserPageFragment.java profile页控件

VideoFragment.java 视频组件

* 1. Utils.java 各文件使用的工具（如转换器）
  2. MainApplication.java 全局信息存储
  3. MainActivity.java切换{Login, App}Activity
  4. LoginActivity.java 登录界面
  5. AppActivity.java 主应用界面

1. res 应用所需的各种素材（非.java文件，内容过多且文件名称已展示对应功能，不具体展开）
   1. drawable 存储各图标（略）
   2. font 存储字体文件

playfair\_display\_sc.ttf 标题文本字体

* 1. layout 存储各页面布局（略）
  2. menu

bottom\_nav\_menu.xml 存储navigation所需item信息

* 1. mipmap 存储背景图片及应用图标
  2. navigation

mobiles.navigation.xml 用于recommend; search; profile这三个页面间的切换

* 1. values

colors.xml 控件颜色

strings.xml 软编码字符串（用于填充至布局而非java文件）

dimens.xml 控件尺寸

styles.xml 控件样式

themes.xml 主题

* 1. xml

backup\_rules.xml 默认生成的备份规则

data\_extraction\_rules.xml 默认生成的数据提取规则

network\_permission\_config.xml 网络权限获取规则（读视频用，非常重要）

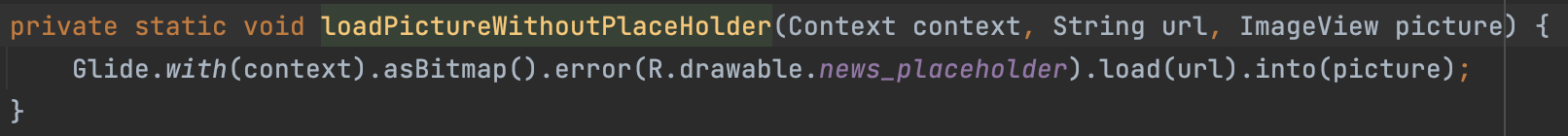
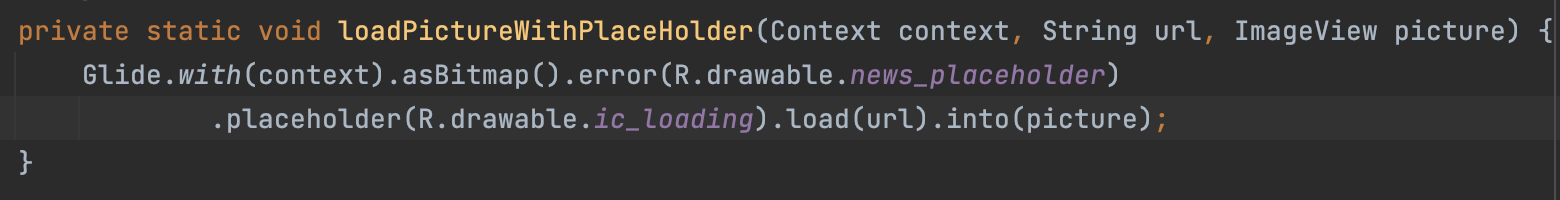
1. **具体实现**
   1. 数据存储与传递

对于新闻数据，本项目综合使用SQLite和xml（android.content.SharePreferences）存储格式。因为涉及持久化大规模新闻文字存储，适合使用SQLite。而既然用了SQLite，复用其代码实现所有存储功能固然很方便，但是SQLite读取速度缓慢，经测试使用SQLite动态加载存储多条新闻时有明显卡顿。而android SDK自带xml存储格式，使用SharePreferences进行交互，在进行少量新闻读取存入时速度很快。经综合考虑，本项目使用“项目启动时从SQLite读取一次历史记录，运行时通过SharePreferences读写文件，每次读取新的新闻时将其写入SQLite（写入速度影响较小）”的策略，实现项目在“Pixel 6 API 28”虚拟机上流畅运行。

对于用户数据，本项目直接使用SQLite存储账户和密码，修改密码功能并未完善（因为没有加入邮箱验证等API），期待日后进一步开发。

* 1. 图片、视频读取

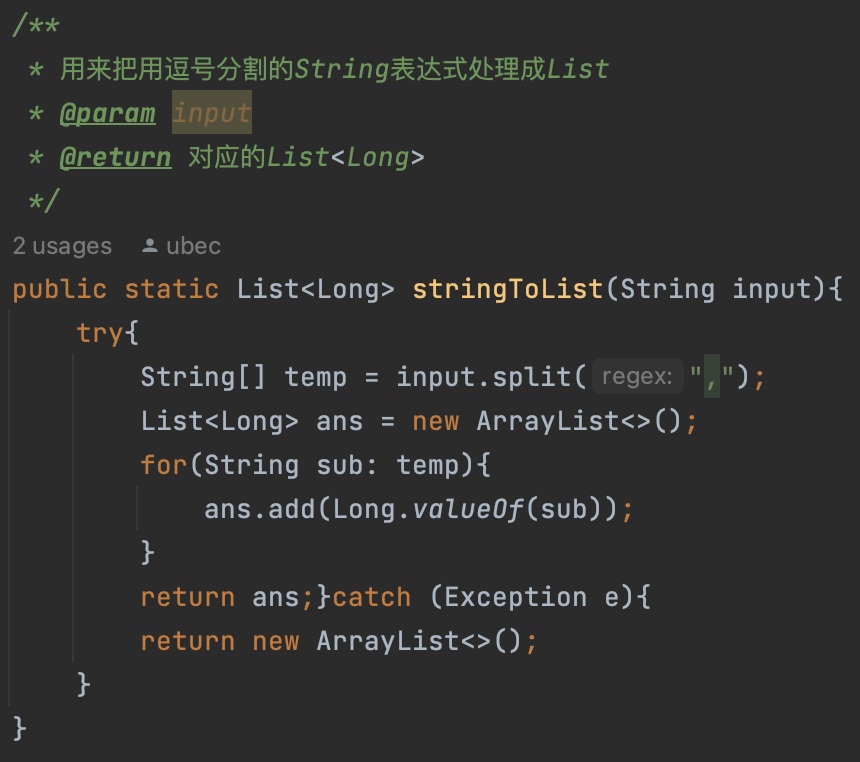
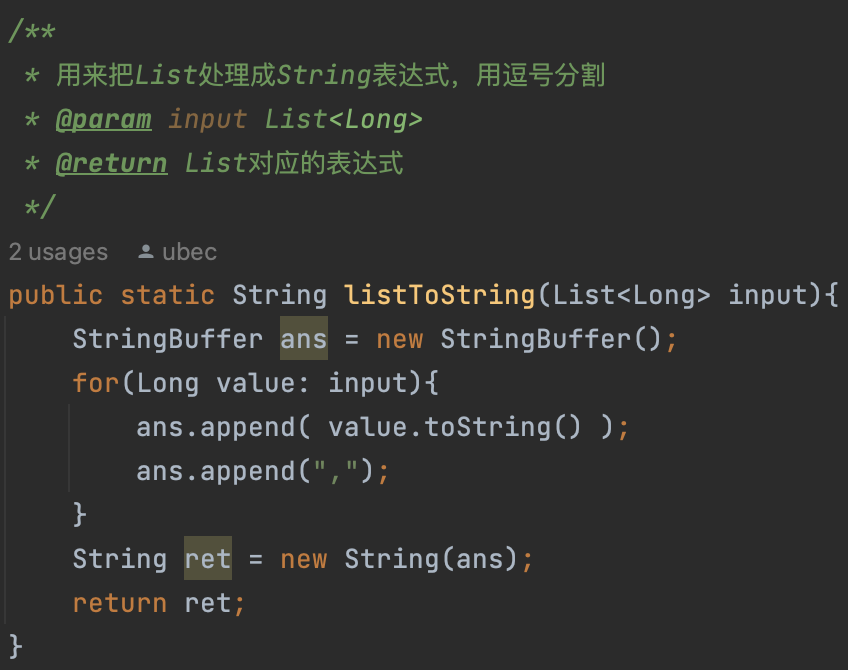
对于图片，本项目使用Glide库，其使用很便捷，但发现Glide库的过程较为曲折，经过很久的搜索…根据是否需要占位符对图片加载进行区分，如下图所示。



对于视频，本项目直接使用了android.widget.VideoView。看上去该组件简单易用，但简单易用的背后隐藏了无数心酸。因为其需要配置AndroidManifest和res/xml，见下“总结与心得”部分。

* 1. 网络请求交互

本项目使用JSONObject和HttpURLConnection库处理GET请求，并由Executor库实现多线程请求（由于涉及信息量较小，实际只使用了newSingleThreadExecutor）。首先，由Executor的实例（位于HttpManager.java）调用APIManager.java中的getNews函数，getNews函数又会调用APIManager.execute()并更新、返回List<News> newsFetched。实际上，APIManager先使用HttpURLConnection获得Response类的response，将response转换为String类型后再用JSONObject将其转换为结构体格式赋值给NewsResponse，后处理为News类型，这一步涉及到了字符串转结构体。而在结构体格式是无法被直接存入SQLite的，故本项目先将结构体转换为字符串格式（见下图），再将字符串存入SQLite，而读取时先读取字符串，逆解析为结构体格式后再使用。



* 1. **登录界面\***

在需求之外，本应用还设计了登录功能。采用SQLite存储，支持注册、登录、修改密码功能（但未加入验证相关API）。而由此使用Intent在Activity间跳转，与Inflater在Fragment间跳转相结合，构成本项目的组件切换功能。

* 1. 新闻展示

本项目使用SwipeRefreshLayout嵌套RecycleView实现任意多条新闻的展示。使用RecycleView.ViewHolder包裹RecycleView，动态填充前者的内容（见NewsListAdapter.java的bindData函数），这是因为RecycleView本身并不可以携带TextView、ImgeView等组件。

对于上划刷新功能，在NewsListAdapter.java里对SwipeRefreshLayout重载setOnRefreshListener函数，每次refresh后执行reloadNews函数即可。而对于加载功能，设置onScrollListener即可实现。在刷新时，还使用了ProgressBar实现了loading状态。这里值得注意的是，由于java纯面向对象的特点，我们可以在NewsListFragment里创建一个NewsListFragment的实例，之后需要调用其他非static类型函数时直接调用NewsListFragment.getInstance().myFunc(args)即可！

* 1. 界面布局

应用图标使用了Android Studio自带的 Image Asset，将jpm文件转换组合形成不同像素的png文件。

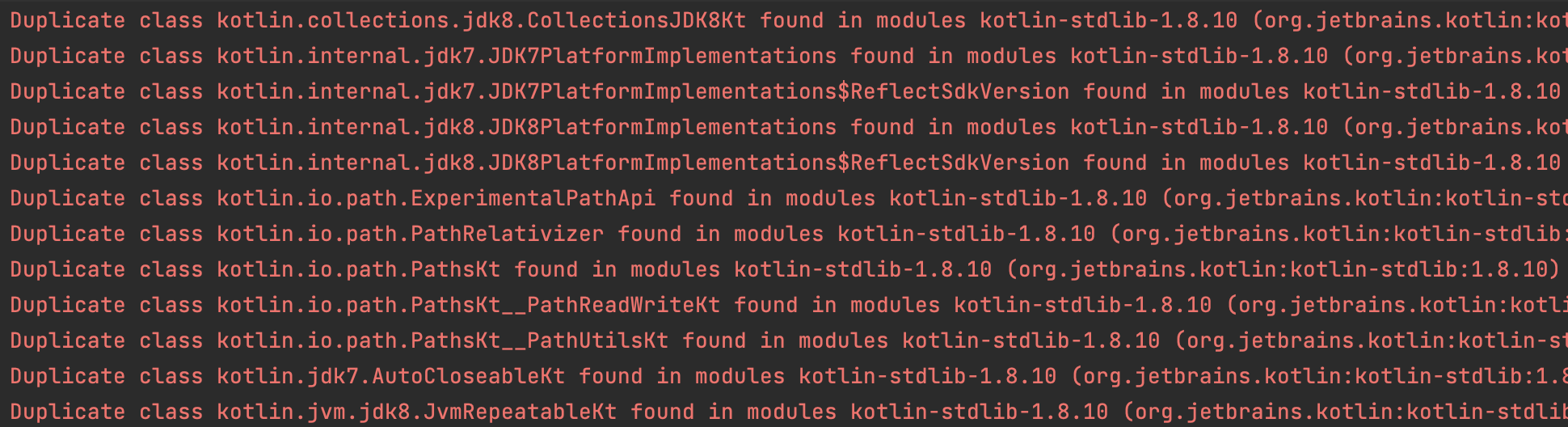
由于jpg、png等格式图片大小固定，为了应用适应性，本项目使用Android Studio自带的Vector Asset将svg格式图片转换为xml格式，嵌入到布局中。

关于具体布局文件。我起初给recommend/profile/search页面分别写了navagation menu，但后来发现可以只写一个，然后在这三个布局当中分别调用。这样做还可以像文字“软编码”一样，需要改显示时只需修改一处。

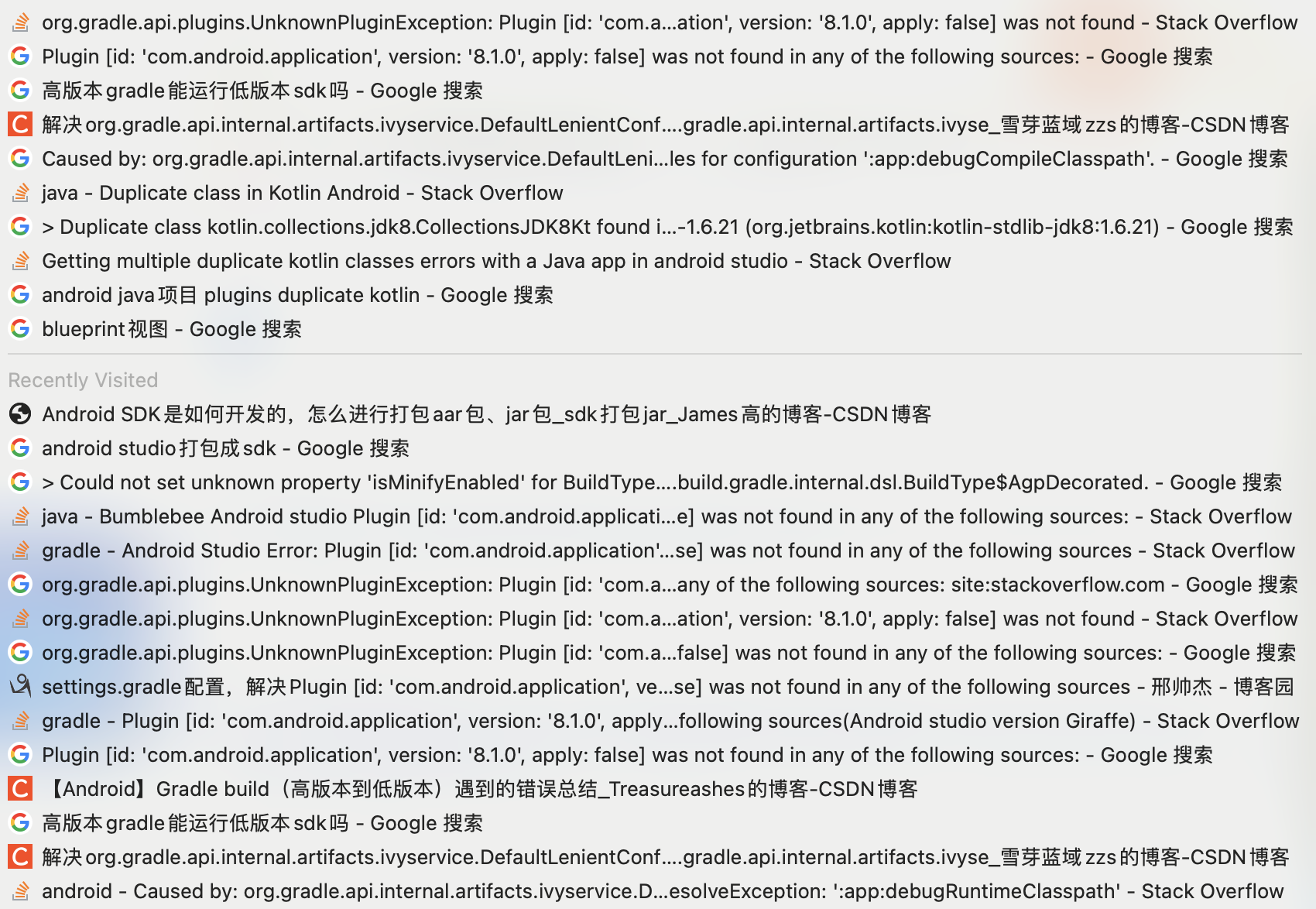
另外，一定要区分好LinearLayout、RelativeLayout、ConstrainLayout等不同布局模式，其实现的功能是不一样的。比如前者适合一行或一列摆很多相同的Button，而后两者就不适合这两种场景。如果搞不清楚的话会做不出布局，因为Android Studio的布局操作还没有智能到拖动即可（现有的拖动是固定绝对px，不适合跨平台使用），而摆放元素也给我造成了巨大的麻烦…

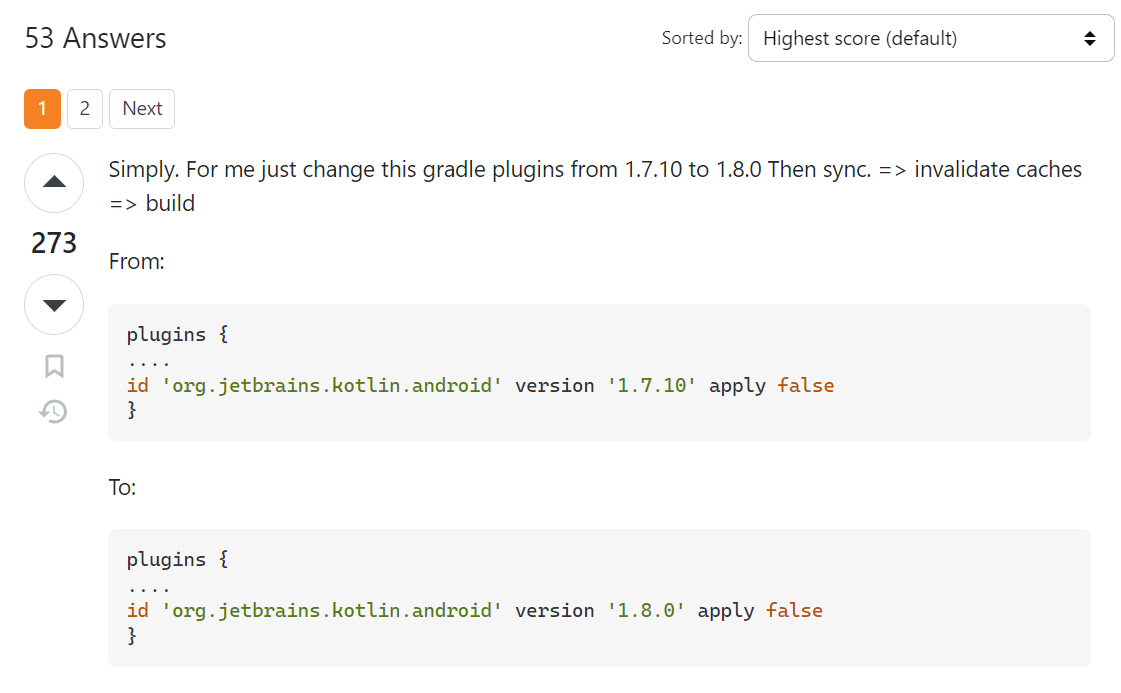
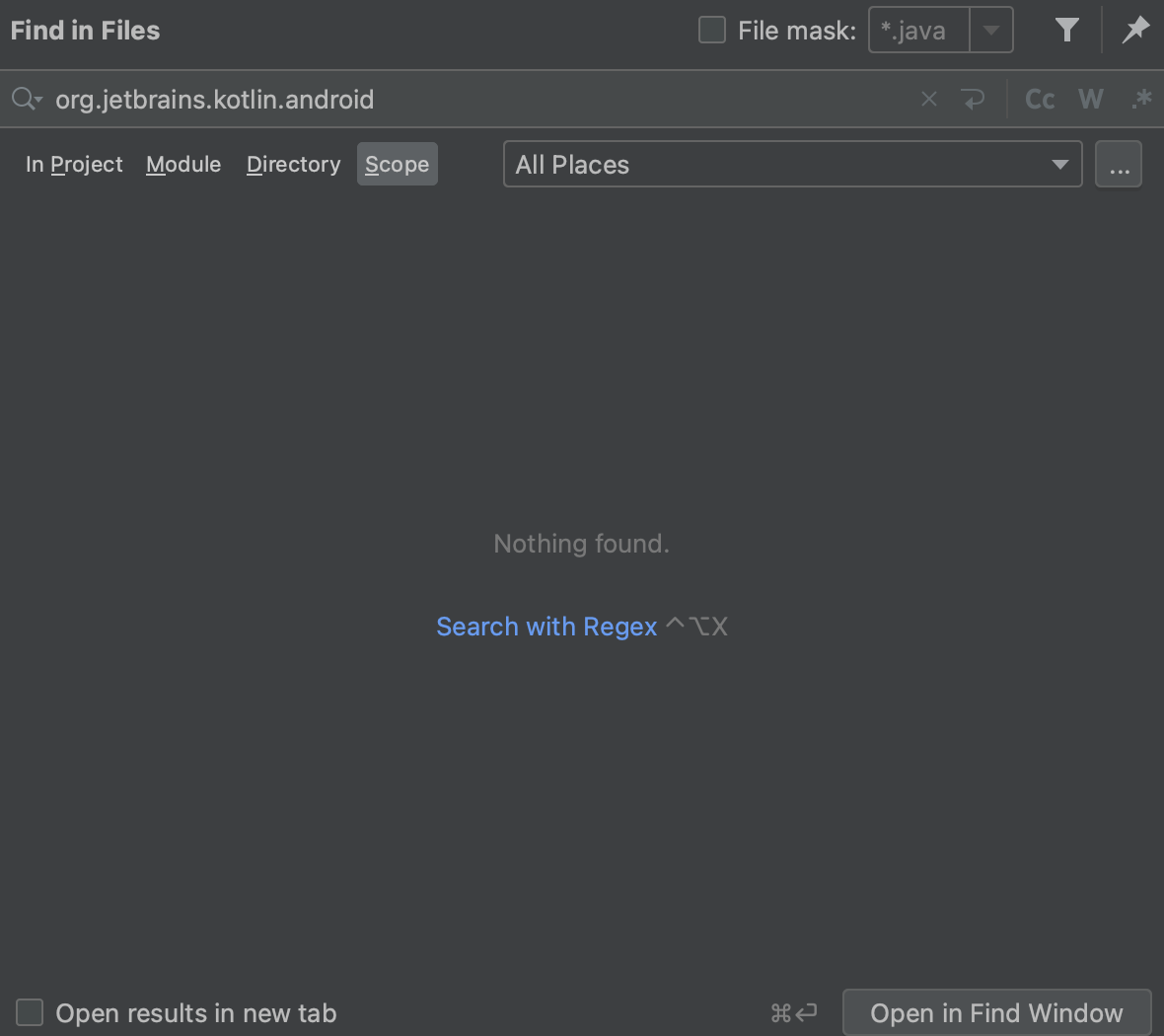
1. **总结与心得**
   1. 环境配置

在成功build项目前，我经历了一段痛苦的回忆，这是从Duplicate报错开始的…

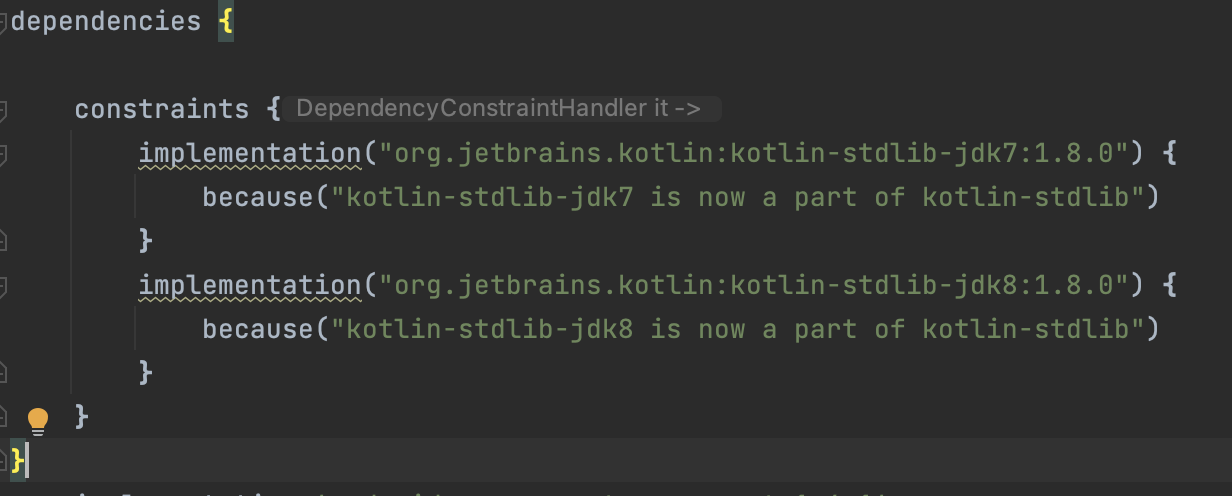


在这之后，我开始问ChatGPT和Google，但得到的结果都无法解决我的问题…



折腾这个问题接近三个小时后，我终于找到了一篇靠谱的答案。这是因为新版本的jdk把kotlin-stdlib-jdk{7,8}删除了，但老版本的android库仍然在使用引用该库，多次安装导致了冲突，解决方法是在项目开始前自己先把jdk{7,8}装好，这样其他库就不会反复安装它们了，项目build成功的一刻感觉要吐血了。感谢上帝，赐予我StackOverflow上的这条回复！



这次的收获是：碰上报错不要着急，也不要查到一个解决方案直接去试，先主观判断一下这个方案是否靠谱，否则会浪费很多时间

* 1. 第三方库的使用

在遇到lombok库之前，我还在傻傻的手写getter和setter还有constructor，而这些代码只是在锻炼我的手速，并没有任何思维性工作。在我遇到lombok库之后，我发现用轮子真爽，这也使我的代码变得清爽很多。

很多功能早已有对应的第三方库实现，我们可以于实现前在搜索引擎上查找是否有相关库存在，了解其实现原理后直接调用有时是非常棒的选择。当然，选择纯手写对自己的锻炼是更大的，但机械性的工作可以使用第三方工具完成。

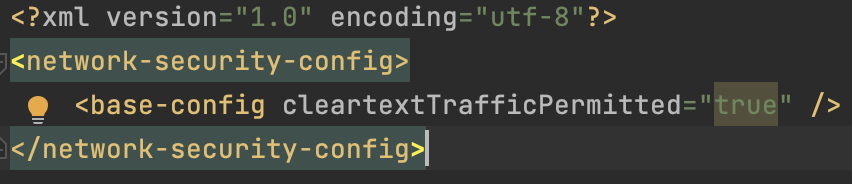
* 1. 报错信息获取

进行安卓开发时，Log.d是必不可少的工具，而Log的命名又是一门学问。我刚开始时，给所有的Log.d对应的Logger都设置成”Logger”这一名字，而报错信息直接用函数名代替。等项目写到后期时，一打开报错发现一大堆”Logger”, “onCreateView”，根本分不清谁是谁的。而将这一大堆Log信息修改又是一件很麻烦的事。所以我领悟到了把Logger名字设置成组件名，报错信息里具体加一些参数和位置，比如我在某个文件里调用了三次func函数，那么我在其前面分别加Log.d(“componentName”, “func {1, 2, 3}”)是最妥当的选择。

不过全命名成一种也有一个好处，就是方便在浩如烟海的Log信息里针对功能而不是组件查找你想要的那几条报错。比如你正在调试API组件，同时需要HttpManager和APIManager的报错信息，而其他的报错信息又很杂。你只需要把Logger的名字设置成功能名，然后在Logcat的filter里输入功能名即可筛选。

* 1. 视频播放功能设置

我按照网上教程设置好VideoView并在AndroidManifest里写入user-permission后，发现项目总是显示“播放视频失败”，而Logger也告诉我播放错误是（-1，2147483648），翻译成人话就是“未知错误”。我尝试套ChatGPT和CodeGeeX的话，但它们都没有给我合理的解决方案。最后又是两小时的搜索加尝试，发现还需要在<Application>标签里设置networkSecurityConfig属性，而这个属性的值并不是true/false，你还需要手写一个xml文件。



类似3.1，在视频运行起来后我想再吐一口血…这件事给我们的启示是，遇到奇怪的报错（尤其是Unknown Error）不要害怕，耐心慢慢尝试解决，着急只会使你更难解决问题。

1. **致谢**

感谢许老师，感谢诸位助教。感谢ChatGPT、Google、ChatGLM、CodeGeeX等一种工具，在你们的帮助下，才有了这个项目，也使我在我的第一个项目中体会到了迭代开发的感觉！