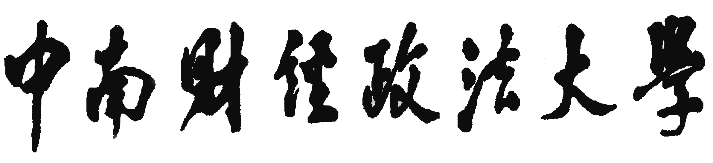
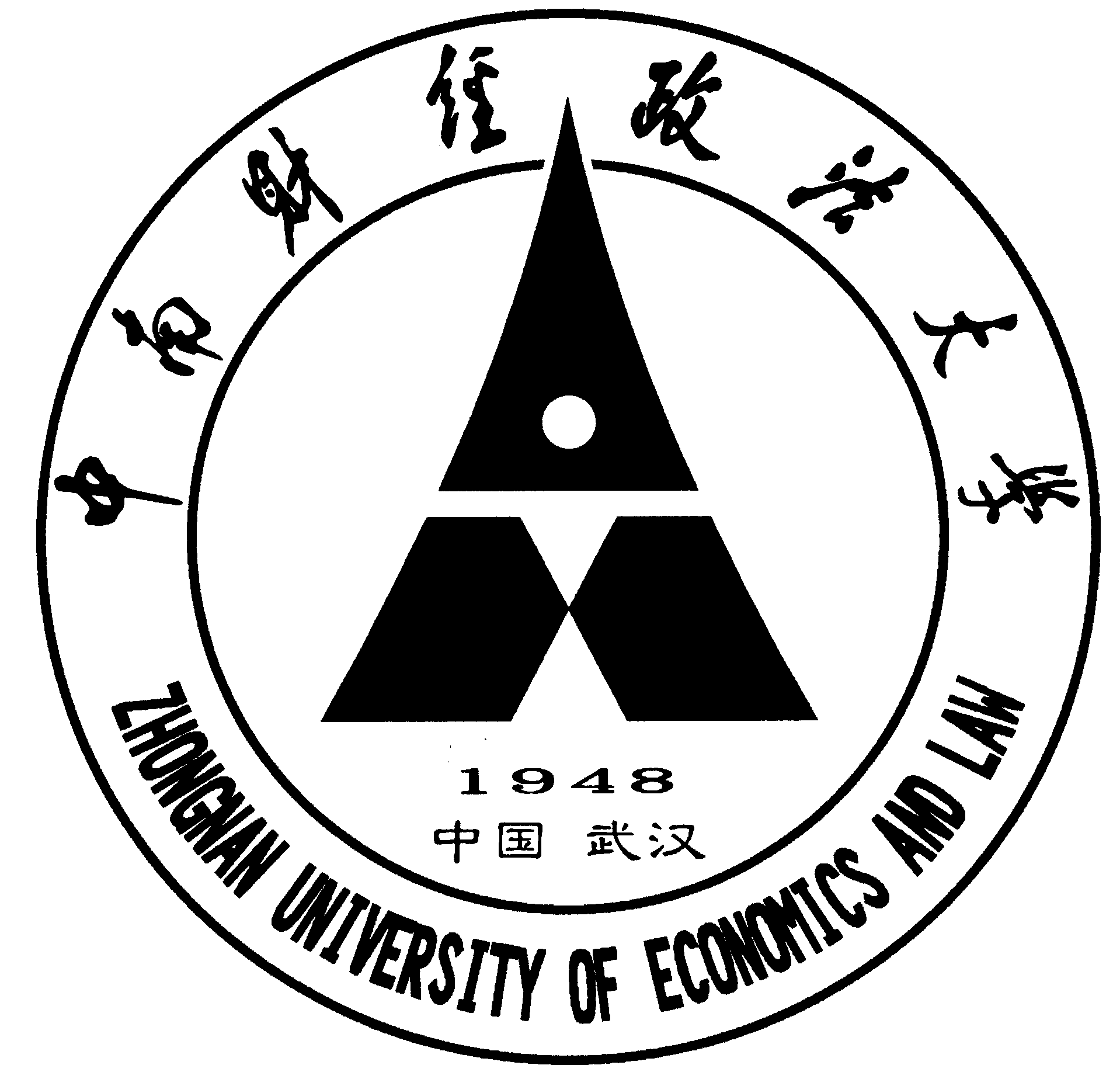
****

**嵌入式开发**

2019-2020学年 第一学期



|  |  |  |
| --- | --- | --- |
| 项目名称 | **：** | **Vision Translation翻译软件开发** |
| 姓名 | **：** | **赵文 付康 夏谦** |
| 学号 | **：** | **201721130272 201721130264 201721130283** |
| 年级 | **：** | **2017级** |
| 专业 | **：** | **计算机科学与技术** |
| 指导教师 | **：** | **孙夫雄** |
| 完成时间 | **：** | **2019 年 10 月 27日** |

**目 录**

**一、 项目背景**………………………………….……………………………….....3**二、 技术基础理论** ...……………………………………….…………………....3

（一）Android系统简介……………………………………………………………….…………………..…..3

（二）Android应用程序构架………………………………………………………………………..………...4

（三）百度翻译接口和google 神经网络翻译模块……………………..……………………………………4

（1）百度在线翻译……………………………………………...……………………………….…….4

（2）Google离线翻译…………………………………………………………………………….…...5

（四）科大讯飞语音转文字Adroid SDK………………………………………………………………...5

（五）Google文字转语音引擎 ………………………..…………………………...…………………….6

**三、 项目设计**…………………………………………………………………….6

（一）需求分析……………………………………………………………….…………………………...6

（二）APP运行流程图 ………………………………………………....…………………………….7

**四、 项目实现**…………………………………………………………………….7

（一）文本翻译……………………………………………………………….…………………………...7

（二）语音播报…………………………………………………………………………………………...16

（三）语音翻译…………………………………………………………………………………….……..17

（四）拍照翻译…………………………………………………………………………………….….….20

**五、 成果展示**………………….………………………………………………...26

（一）APP运行主界面文本翻译模块…….……………………….……………………………….…….26

（二）拍照翻译模块………………………………………….…………………………………………..27

（三）语音翻译模块……………………………………….……………………………………………..29

（四）菜单模块………………………………………………….………………………………………..30

**六、 总结**…………………………………………………………………………32

(一) 项目成果.………………………………………………………………………………………….32

(二) 小组分工……………………………………………………………………………………………….32

(三) 收获…………………………………………………………………………………………………….32

1. **项目背景**

目前市场上，单词文字翻译，语音识别翻译的软件很多，甚至我们只需要上百度就可以直接查单词，识别语音内容，但是当我们遇到一种自己不认识，不会读并无法拼写出来的语言文字时，我们就无法去查这些文字的意思，为了使翻译更通用，更快捷，更方便，我们小组打算做一个可以使用相机实时翻译的软件。用户只需要打开软件，使相机对准文字，聚焦后就可以将文字翻译成我们需要的语言，比如将书上的英文翻译成中文显示在软件相机界面上，便于用户理解文字内容。

本次项目的app是在Android系统上开发的，使用Android Studio平台进行软件开发。做这个项目的优势在于目前文本翻译的API比较多，我们主要任务是抓取相机内容，进行平面追踪，获取到文字后可以直接使用文本翻译模块进行翻译显示在相机上，支持的语言种类繁多。根据我们小组的综合分析，实现app的功能要求是有可行性的。

项目的独特与创新之处是相机实时翻译，遇到看不懂，不认识，拼写不出来的文字而无法查询时，就可以打开我们的这个app，将软件中的相机对准文字，并选择需要翻译成的语言，就可以将翻译显示相机屏幕上，出国旅游人下载我们这款app就不再害怕担心不识字了，而且当我们阅读外文书籍时，也可以使用app的相机实时翻译功能，这样阅读就更轻松。

除此之外，我们还打算添加文本翻译和语言识别翻译功能，当用户需要查单词，语音识别时，也可以直接使用我们的app，另外，我们还将使用Android自带的tts接口，向app添加文本转语言功能，可以将文字读出来，满足用户需要知道文本发音的需求。app功能丰富实用，在翻译软件中具有很强的竞争优势。

1. **技术基础理论**

**（一）Android 平台简介**

Android是一种基于Linux的自由及开放源代码的操作系统，主要使用于移动设备，如智能手机和平板电脑，由Google公司和开放手机联盟领导及开发。尚未有统一中文名称，中国大陆地区较多人使用“安卓”或“安致”。Android操作系统最初由Andy Rubin开发，主要支持手机。2005年8月由Google收购注资。2007年11月，Google与84家硬件制造商、软件开发商及电信营运商组建开放手机联盟共同研发改良Android系统。随后Google以Apache开源许可证的授权方式，发布了Android的源代码。第一部Android智能手机发布于2008年10月。Android逐渐扩展到平板电脑及其他领域上，如电视、数码相机、游戏机等。2011年第一季度，Android在全球的市场份额首次超过塞班系统，跃居全球第一。 2013年的第四季度，Android平台手机的全球市场份额已经达到78.1%。[1]2013年09月24日谷歌开发的操作系统Android在迎来了5岁生日，全世界采用这款系统的设备数量已经达到10亿台。

**（二）Android 应用程序框架**

开发人员也可以完全访问核心应用程序所使用的API框架。该应用程序的架构设计简化了组件的重用;任何一个应用程序都可以发布它的功能块并且任何其它的应用程序都可以使用其所发布的功能块（不过得遵循框架的安全性）。同样，该应用程序重用机制也使用户可以方便的替换程序组件。隐藏在每个应用后面的是一系列的服务和系统, 其中包括以下内容：

1. 视图（Views)：可以用来构建应用程序，它包括列表（Lists)，网格（Grids)，文本框（Text boxes)，按钮（Buttons)，甚至可嵌入的web浏览器。
2. 内容提供器（Content Providers)：使得应用程序可以访问另一个应用程序的数据（如联系人数据库)， 或者共享它们自己的数据
3. 资源管理器（Resource Manager)：提供非代码资源的访问，如本地字符串，图形，和布局文件（Layout files )。
4. 通知管理器 （Notification Manager):使得应用程序可以在状态栏中显示自定义的提示信息。
5. 活动管理器（ Activity Manager): 用来管理应用程序生命周期并提供常用的导航回退功能。

**（三）百度翻译接口和google 神经网络翻译模块**

**（1）百度在线翻译**

1.使用您的百度账号登录百度翻译开放平台（https://api.fanyi.baidu.com）；

2.注册成为开发者，获得APPID；

3.进行开发者认证（如仅需标准版可跳过）；

4.开通通用翻译API服务；

5.参考技术文档和Demo编写代码

6.接入方式如下：

通用翻译API通过HTTP接口对外提供多语种互译服务。您只需要通过调用通用翻译API，传入待翻译的内容，并指定要翻译的源语言（支持源语言语种自动检测）和目标语言种类，就可以得到相应的翻译结果。

**（2）Google 离线翻译**

离线翻译使用谷歌Firebase 机器学习套件中的自然语言处理模块，该模块的实现了语言检测和57中语言的互译，翻译以英语为中间语言。翻译精度比传统的离线翻译略高，基于神经网络的翻译也使得翻译质量优于传统的解决方案。以英语为中间语言也会导致非英语语言之间翻译的质量堪忧，但对于Android这样的嵌入式设备，显著减少了所需要的模型数量和体积，单个语言模型的大小在30MB左右。比起云端翻译，设备端翻译适合于一些以查词为主的应用场景。

**（四）科大讯飞语音转文字SDK**

讯飞开放平台作为全球首个开放的智能交互技术服务平台，致力于为开发者打造一站式智能人机交互解决方案。用户可通过互联网、移动互联网，使用任何设备、在任何时间、任何地点，随时随地享受讯飞开放平台提供的“听、说、读、写……”等全方位的人工智能服务。目前，开放平台以“云+端”的形式向开发者提供语音合成、语音识别、语音唤醒、语义理解、人脸识别、等多项服务。

国内外企业、中小创业团队和个人开发者，均可在讯飞开放平台直接体验世界领先的语音技术，并简单快速集成到产品中，让产品具备“能听会说会思考会预测”的功能。

快速指引如下：

1. 进入讯飞开放平台任意页面 ，点击右上角注册，在弹出的窗口完成账号注册相应步骤，即可成为讯飞开放平台注册开发者。

2. 登录平台后，通过右上角「控制台」，或右上角下拉菜单的「我的应用」进入控制台。若账户未曾创建过应用，平台会引导我们创建我们的第一个应用。为应用起一个名字，并填写相关的信息。点击提交按钮后，应用就创建完毕。应用创建完成之后，我们就可以通过左侧的服务列表，选择我们要使用的服务。在服务管理面板中，我们将看到这个服务对应的可用量、历史用量、服务接口的验证信息，还有可以调用的API和SDK了。

3. 如果应用需要SDK方式的接入（包括Android，iOS，Linux等），我们可以通过服务管理页SDK版块的「下载」，点击进入SDK下载页，通过「文档」，查阅开发文档。

4. 如果应用需要API方式的接入，我们可以通过服务管理页API版块，查看具体的调用接口，并通过「文档」，查阅开发文档。

科大讯飞语音听写，是基于自然语言处理，将自然语言音频转换为文本输出的技术。语音听写技术与语法识别技术的不同在于，语音听写不需要基于某个具体的语法文件，其识别范围是整个语种内的词条。在听写时，应用还可以上传个性化的词表，如联系人列表等，提高列表中词语的匹配率。

**（五）Google文字转语音引擎**

文本转换语音技术，一直是Android的辅助功能之一，在Android设备上，都预装了文本转换服务，目前市面上有Google文本转换语言引擎，讯飞文本转换引擎。相比其它引擎Google文本转换语言引擎以其全面覆盖的语音支持，设备和云端共同服务提供方式，在为数不多的文本转换语音引擎中独树一帜。

**三、项目设计**

**(一)需求分析**

**1. 环境需求**

（1） android 操作系统7.0版本及以上

**2.功能需求**

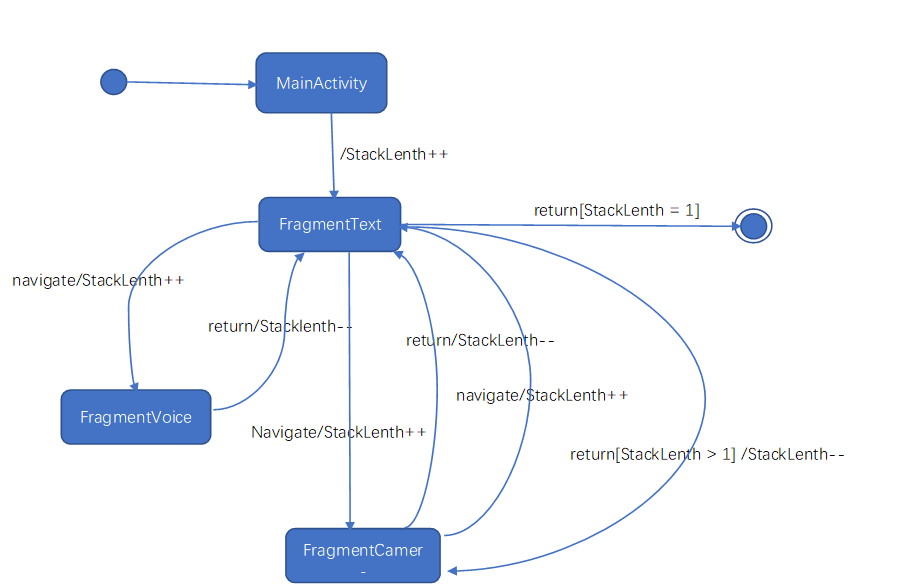
（1）文本翻译：输入一段文字，可以将文字翻译成用户选择的语言的文字，输出到UI上显示，支持“中英日韩”等世界主流语言文字的翻译，添加历史记录和收藏功能，用户可以查看自己的搜索记录，还可以将某个翻译记录收藏，方便用户之后的使用，当然这些历史记录也可以删掉，收藏记录也可以取消。

（2）实时翻译：使用app内的相机聚焦需要翻译的文字，可识别相机界面上的文字，并翻译成用户选择的语言文字。

（3）语音识别+翻译：用户说一段话，可识别出这段话的文字，并可以将文字翻译成用户所选的语言文字。

（4）文本转语言：用户输入的文字和翻译过来的文字都可以转换成语言播放出来。

**（二）APP运行流程图**



**四、项目实现**

**（一）文本翻译模块**

文本翻译采用Firebase 机器学习套件和百度翻译Web API。在需要快速以接近实时的效能时，我们采用Firebase 机器学习套件，和翻译缓存系统来满足速度需求。在需要翻译精度的场景下采用云端翻译API。

在本应用中我们将翻译服务进行封装，留出公共请求接口，结果回调接口来提供翻译服务。翻译服务包括三个部分，翻译API独立封装模块，翻译缓存模块，翻译服务提供模块。

**百度在线翻译模块**：

获取百度翻译API服务后，从HTTP接口发送翻译请求。

通用翻译API HTTP地址：

http://api.fanyi.baidu.com/api/trans/vip/translate

通用翻译API HTTPS地址：

https://fanyi-api.baidu.com/api/trans/vip/translate

编码实现HTTP GET请求方式，需要的参数有请求翻译的字段q，翻译源语言from（设置为auto可自动识别），译文语言to（不能设置auto）， APP ID（appid可在管理控制台查看），随机数（salt），签名。签名sign是appid+q+salt+密匙的MD5值。此时，要编写代买实现MD5算法，获取签名。

语言列表如下：

|  |  |
| --- | --- |
| **语言简写** | **名称** |
| auto | 自动检测 |
| zh | 中文 |
| en | 英语 |
| yue | 粤语 |
| wyw | 文言文 |
| jp | 日语 |
| kor | 韩语 |
| fra | 法语 |
| spa | 西班牙语 |
| th | 泰语 |
| ara | 阿拉伯语 |
| ru | 俄语 |
| pt | 葡萄牙语 |
| de | 德语 |
| it | 意大利语 |
| el | 希腊语 |
| nl | 荷兰语 |
| pl | 波兰语 |
| bul | 保加利亚语 |
| est | 爱沙尼亚语 |
| dan | 丹麦语 |
| fin | 芬兰语 |
| cs | 捷克语 |
| rom | 罗马尼亚语 |
| slo | 斯洛文尼亚语 |
| swe | 瑞典语 |
| hu | 匈牙利语 |
| cht | 繁体中文 |
| vie | 越南语 |

生成MD5算法加密的签名后，拼接完整请求：http://api.fanyi.baidu.com/api/trans/vip/translate?q=apple&from=en&to=zh&appid=2015063000000001&salt=1435660288&sign=f89f9594663708c1605f3d736d01d2d4

返回的结果是json格式：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **类型** | **描述** | **备注** |
| from | TEXT | 翻译源语言 | 返回用户指定的语言，或自动检测的语言（源语言设为auto时） |
| to | TEXT | 译文语言 | 返回用户指定的目标语言 |
| trans\_result | MIXED LIST | 翻译结果 | 返回翻译结果，包含src 和 dst 字段。 |
| src | TEXT | 原文 |  |
| dst | TEXT | 译文 |  |
| error\_code | Int32 | 错误码 | 仅当出现错误时显示 |

其中error\_code只在翻译出错时返回

为了获取json中的内容创建一个符合json内容格式的类：

**class** TranJson{

**public** String from;

**public** String to;

**public** Map[] trans\_result;

}

再使用json包中解析函数得到一个TranJson类的实例，然后就可以读取到trans\_result中的翻译结果。

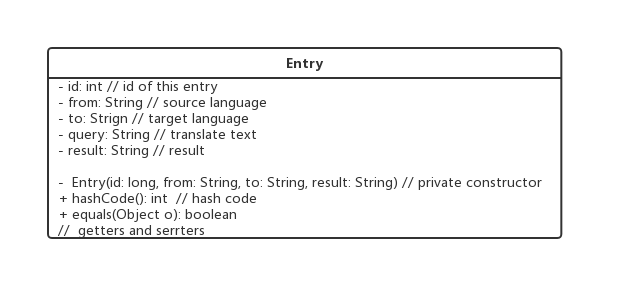
**翻译缓存模块**

在本应用在以TranslationCache类作为翻译缓存的服务的提供者。其提供数据库读写操作，和内存数据缓存服务。

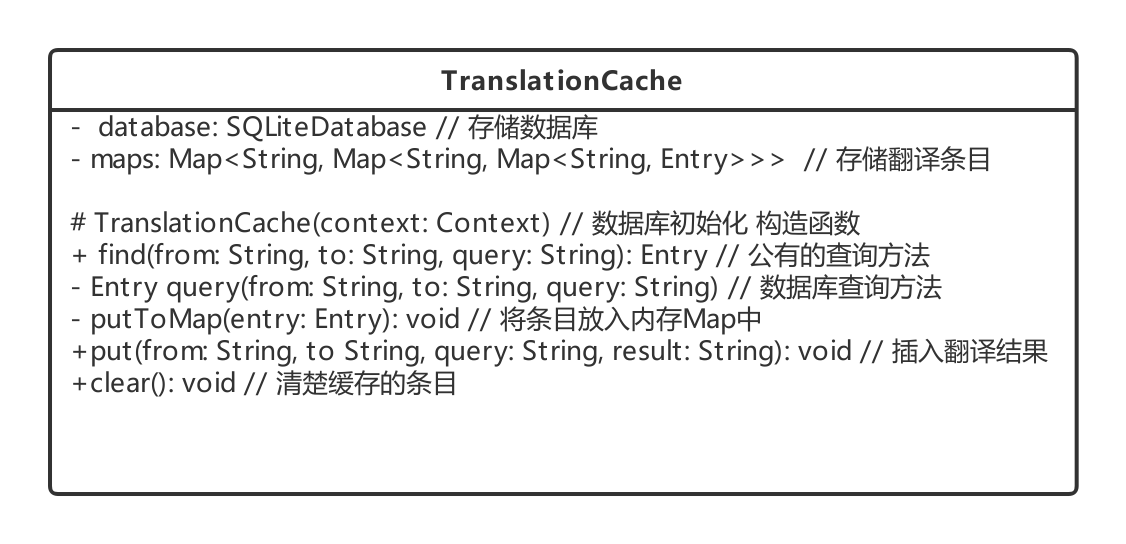
数据表cache结构如下。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 主键 | 非空 | 说明 |
| id | number | yes | yes | 缓存项ID |
| source | text |  |  | 源语言 |
| target | text |  |  | 目标语言 |
| value | text |  |  | 翻译文本 |
| result | text |  |  | 翻译结果 |

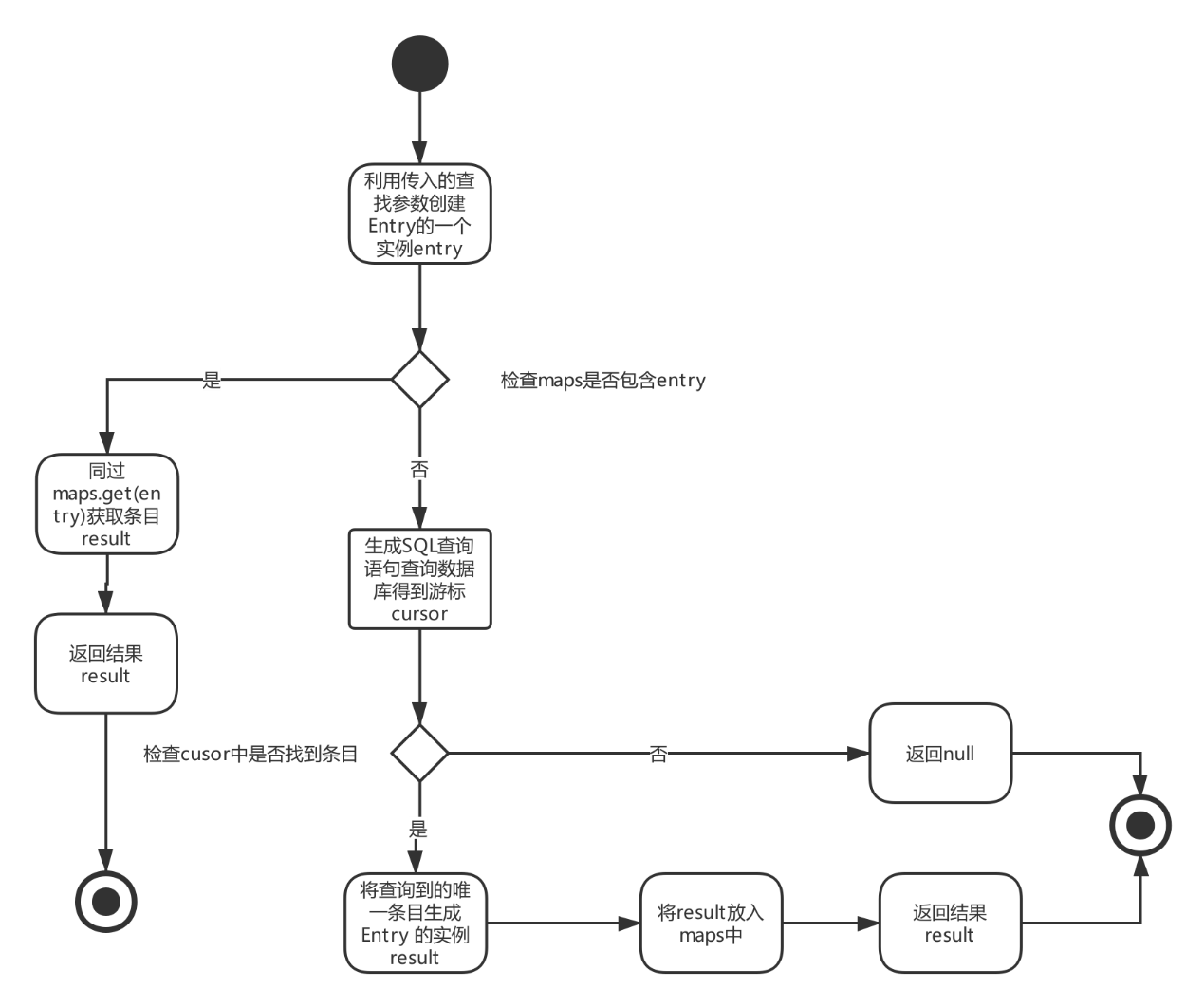
数据内存模型如下，这里重写了hashCode() 和equls() 方法，确保在查询时具有相同的source, target, value 的对象被视为同一对象。



TranslationCache留出了三个个公共方法put(), clear() 和 query()给被调用者使用，分别提供加入缓存项，清除缓存，和查询缓存。TrancslationCache类图如下。

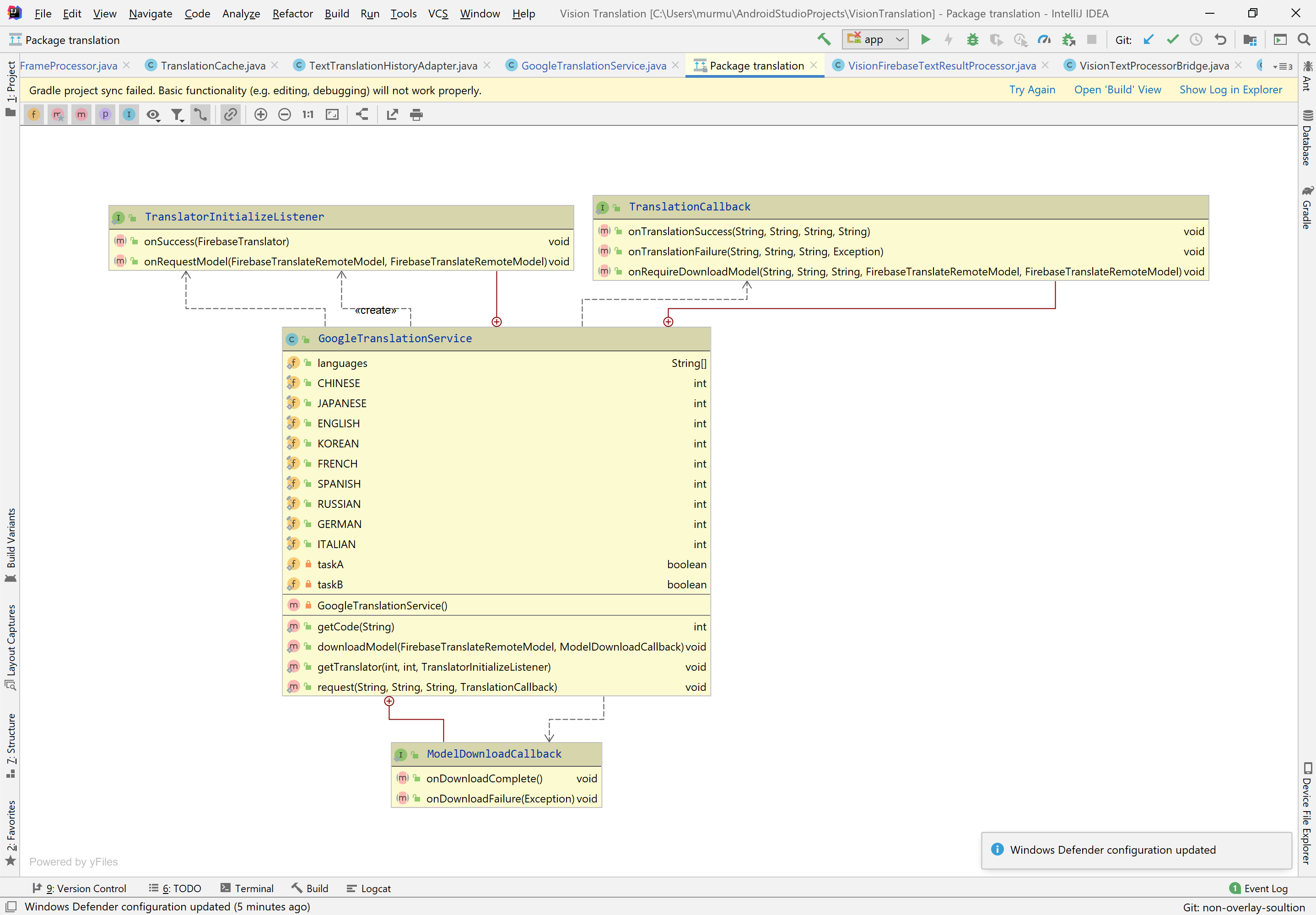


在查找时TransaltionCache的处理流如下



**Firebase Natural Language Translation模块**

该模块实现类是GoogleTranslationServices， 提供了翻译请求，结果回调，模型下载的接口。类图如下。



在翻译通过过静态方法request请求调用，传入参数和翻译结果回调，当翻译完成后，结果会在回调中返回。

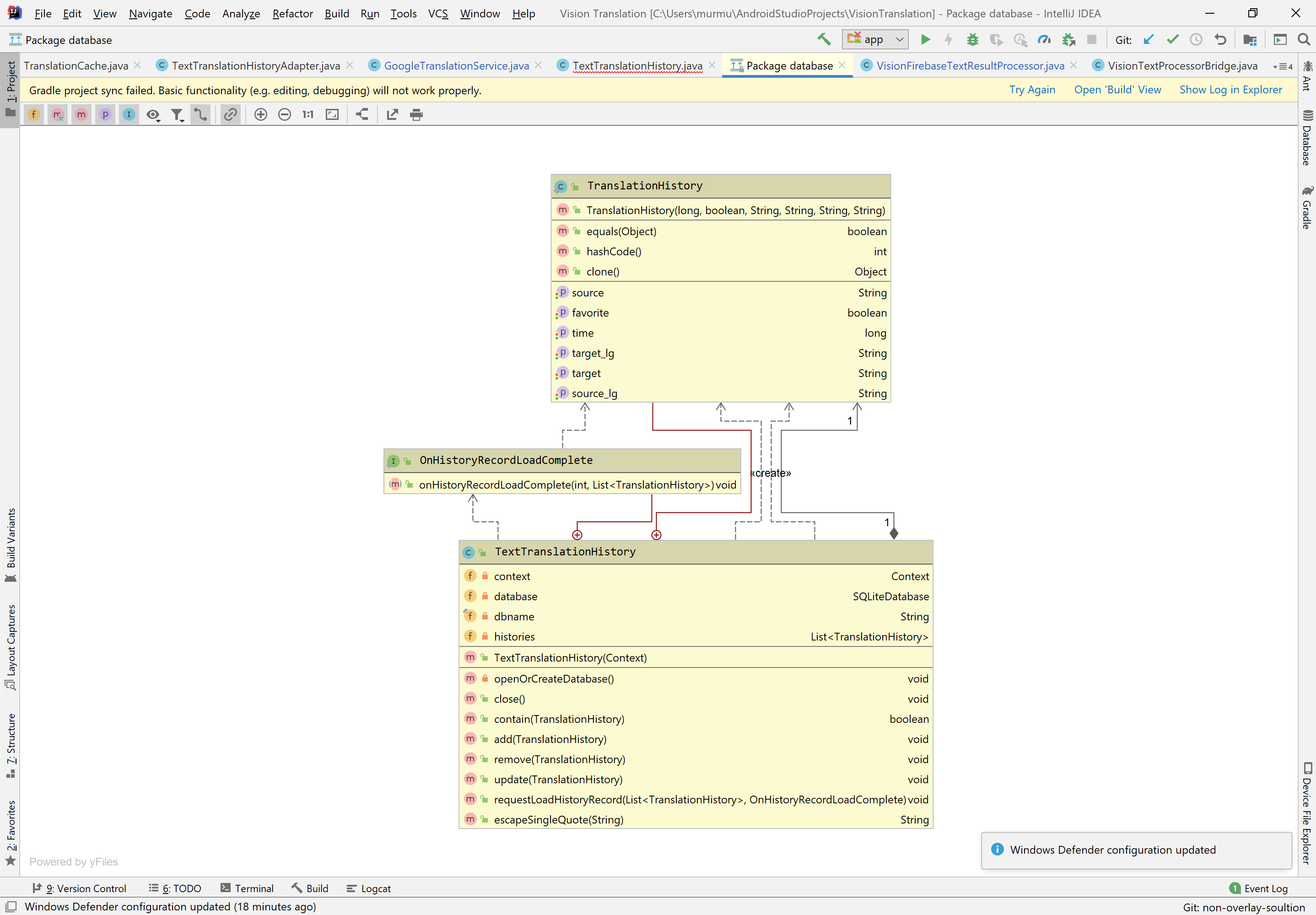
**翻译历史记录模块**

与翻译缓存类似，历史记录采用类似的接口提供查询，更新，修改和删除历史记录条目的功能，增加了异步数据的接口。翻译历史记录的实现类是TextTransaltionHistory。

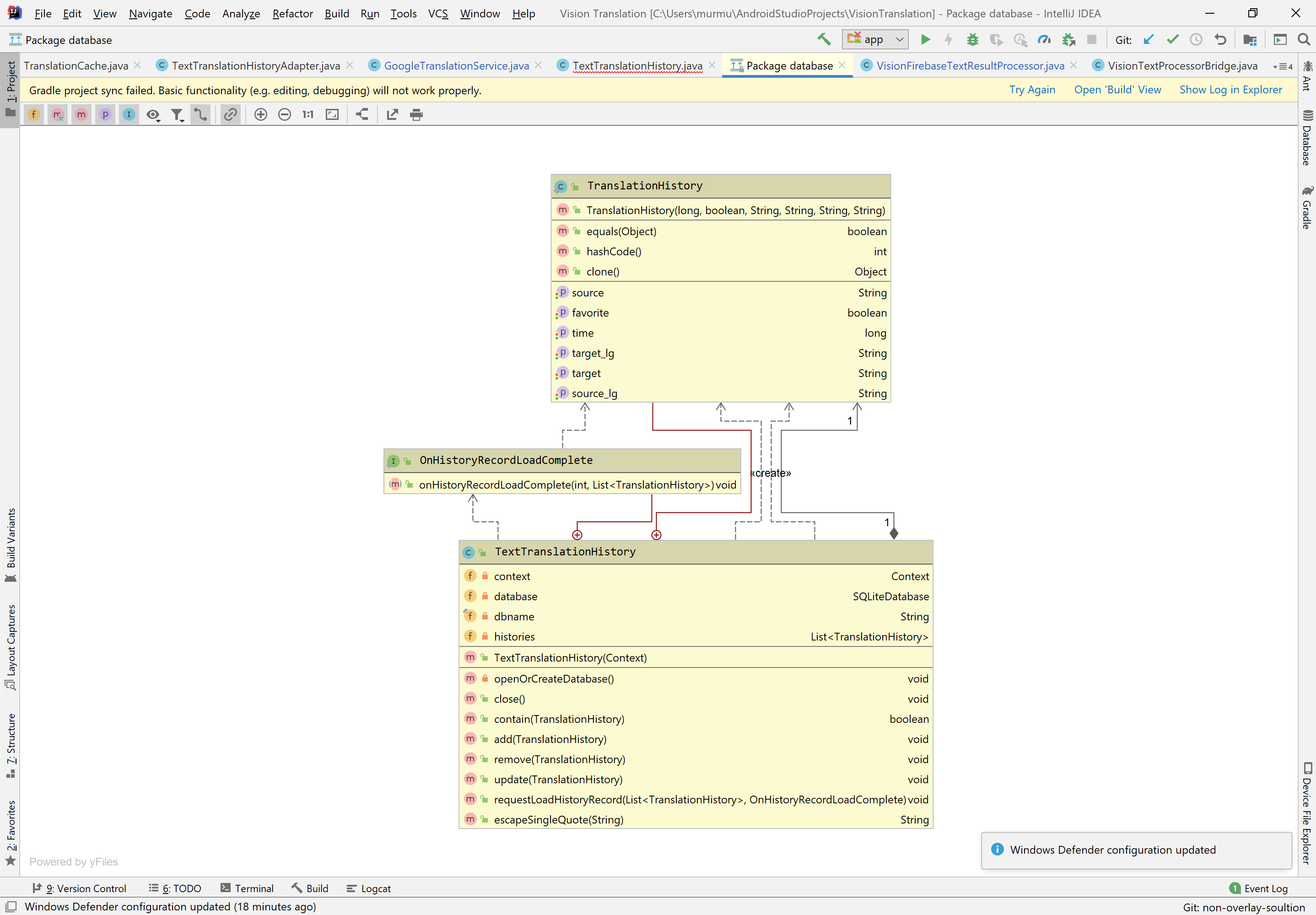
数据库中表结构如下。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 列名 | 类型 | 主键 | 非空 | 说明 |
| source | text | Yes |  | 源语言 |
| target | text | Yes |  | 目标语言 |
| value | text | Yes |  | 翻译文本 |
| result | text |  |  | 翻译结果 |
| time | number |  |  | 记录时间 |
| is\_star | number |  |  | 是否星标 |

记录内存模型如下，这里重写类hashCode和equal()以确保，当两个TranslationHistory对象实例中的source, target, source\_lg, target\_lg成员值都相同时被视作同一实例。从而确保数据的一致性。



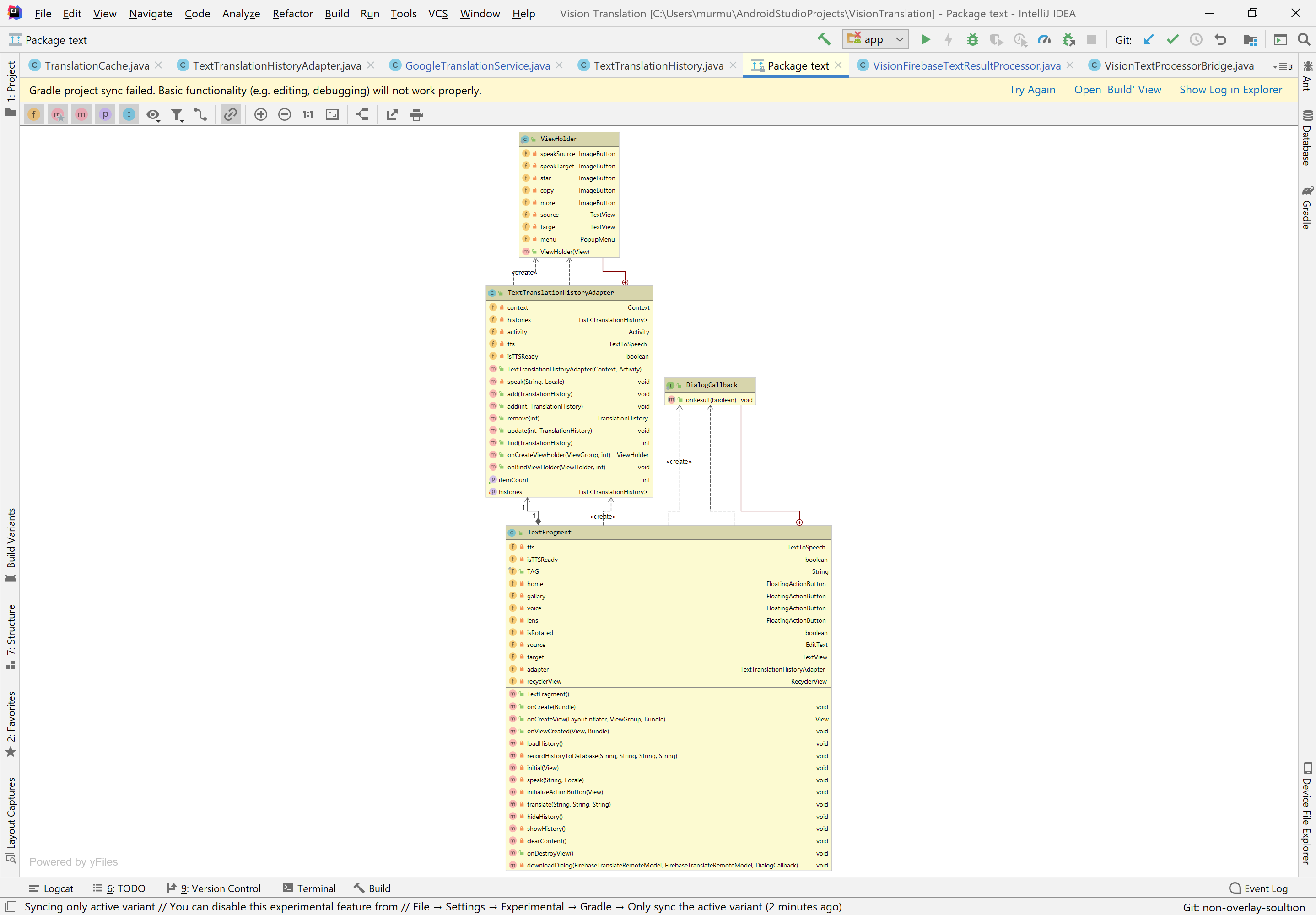
TextTranslationHistory 的类图如下。其活动记录与TranslationCache基本相同。为了保护数据，所有从TranslationHistory 的到历史记录条目都会被复制一份，而返回克隆的条目。TextTransaltionHistory 外部修改TranslationHistory 不会影响到历史记录的缓存，以及数据表。更新数据需要单独调用update()方法，这一点确保了内存数据和数据库数据的一致性。



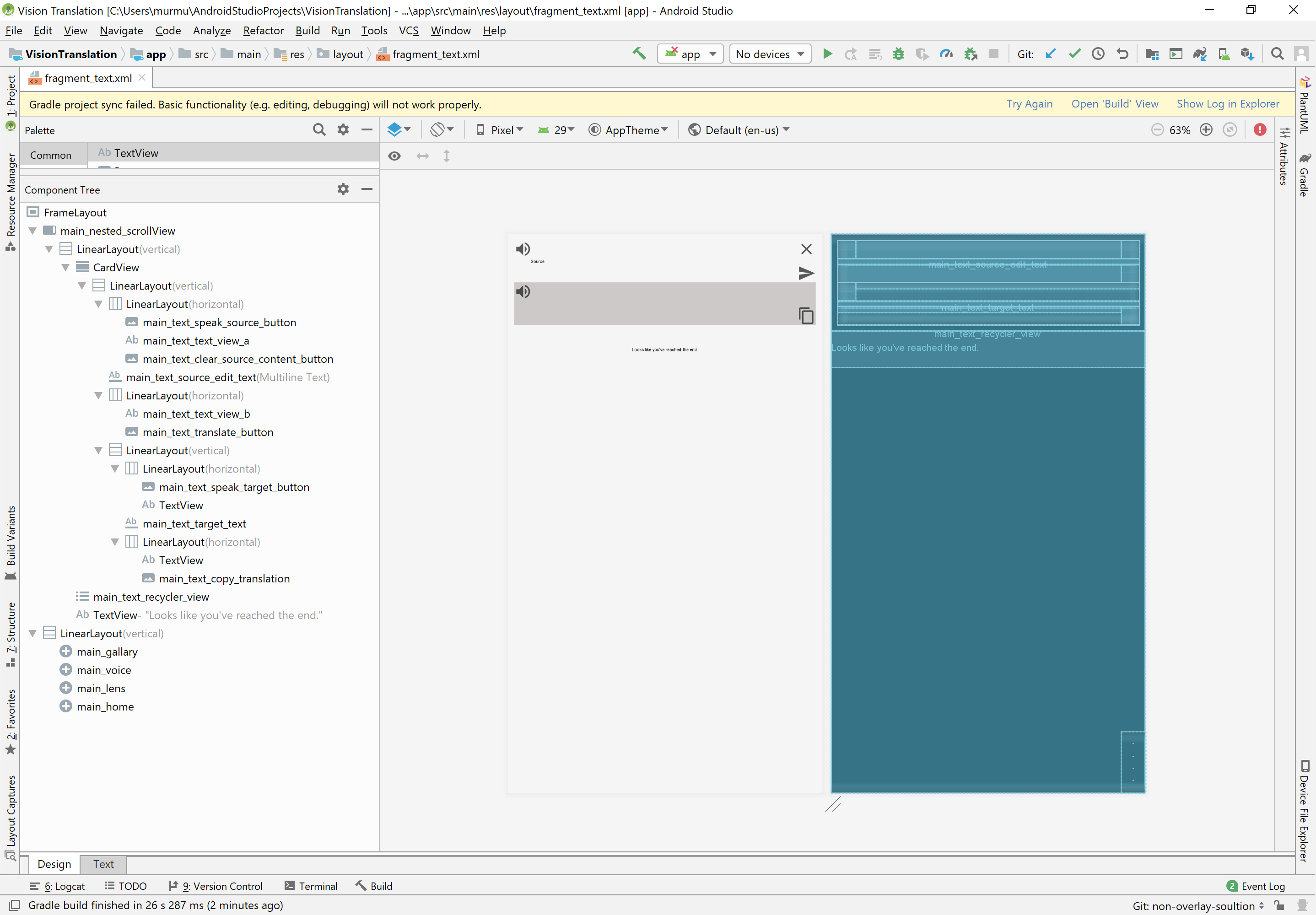
**文本翻译交互模块**

交互界面在依附在TextFragment上，界面和事件处理不再赘述。

主要类图如下。



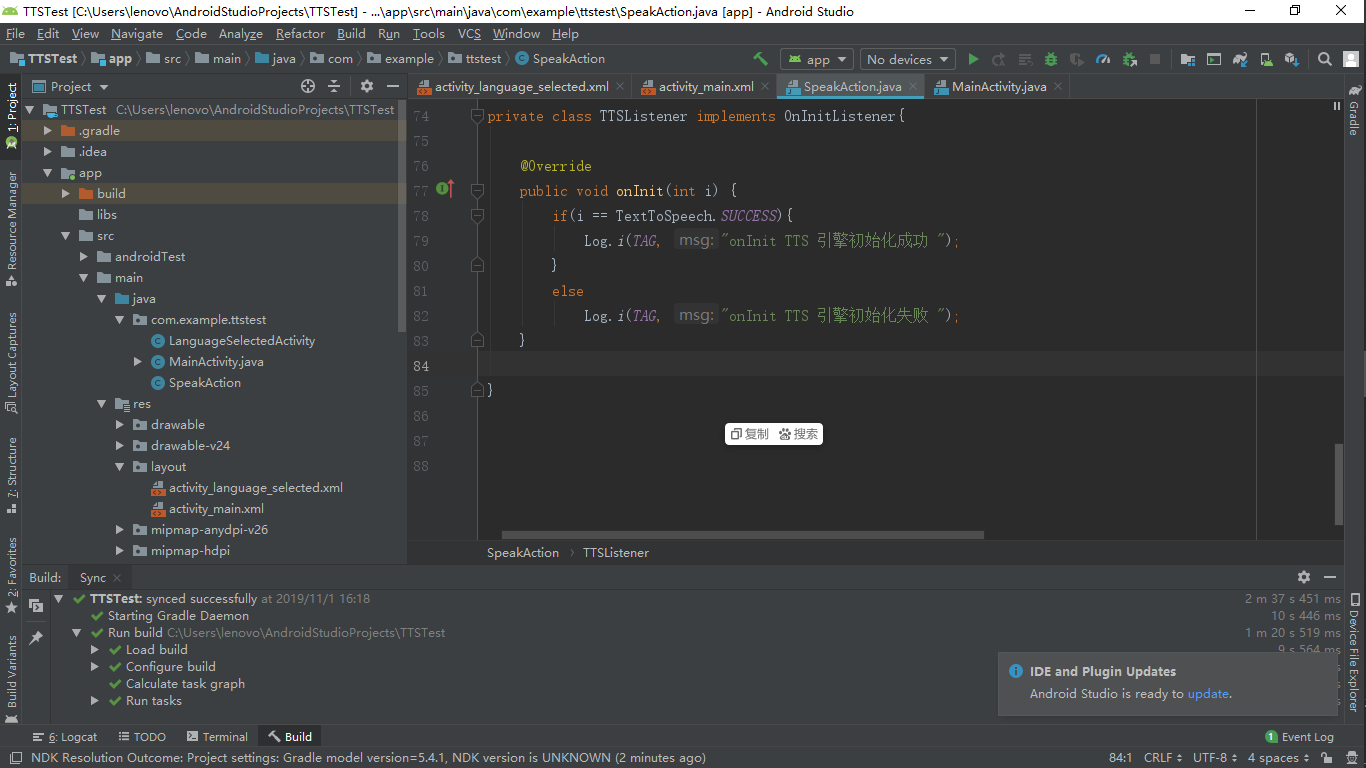
**界面XML布局如下**



**（二）语音播报模块**

Text to Speech即从文本到语音，让机器能够说话，使用Android系统speech包中自带的tts接口，创建初始化TextToSpeech对象时，不指定tts引擎就会使用手机的默认引擎，也可以自己去下载Google文本转语音引擎，在tts初始化指定引擎的包名就可以连接了，可以通过TextToSpeech.getEngines查看有关已安装的文本语音转换引擎的信息。

实现TextToSpeech.OnInitListener接口



初始化TextToSpeech实例：

*mSpeech* = new TextToSpeech(context, new TTSListener());

TextToSpeech.getLanguages()方法可获取tts引擎支持的语言种类

然后调用TextToSpeech.setLanguage()方法设置语音的语言TextToSpeech.setVoice()方法设置语音的特点

调用TextToSpeech.speak（）方法传入文字参数可以播放语音

mSpeech.speak(str,QUEUE.FLUSH,null)，其中str为要读的文字。

**（三）语音翻译模块**

首先需要下载SDK包，将其相应文件导入项目中，添加依赖和相应的权限后即可视为配置成功，可以编写相应的实现代码了。

程序界面如下：



界面为一个LinearLayout布局内嵌套了一个RecycleView布局、TextView、和一个LinearLayout布局。TextView用来显示按下按钮时的提示文字，内部的LinearLayout布局用来存放2个按钮。

RecycleView中很多的子项，子项布局是一个外部LinearLayout嵌套2个内部LinearLayout，2个内部LinearLayout分别存放英文和中文的气泡。内部LinearLayout包括TextView和ImageButton，分别用来存放文本和右下角的音标按钮。

功能实现如下：

1.语音转文字功能：我使用的是无UI识别的在线语音转文字，需要先配置好下载的SDK包，需要注意的是首先需要在开头初始化SDK，即输入我在科大讯飞创建软件是申请的APPID，这里建议放在主线程的开头。拿录入英文举例，主要代码如下：

**mIat**.setParameter(SpeechConstant.***RESULT\_TYPE***, **"json"**);  
*//表示语音解释方式为json***mIat**.setParameter( SpeechConstant.***ENGINE\_TYPE***, **"cloud"** );  
*//cloud表示在线语音转文字***mIat**.setParameter(SpeechConstant.***LANGUAGE***, **"en\_us"**);

*//设置语音输入语言，zh\_cn为简体中文*

**public void** onResult(RecognizerResult recognizerResult, **boolean** b) {  
 **if** (b) {  
 **result** = parseVoice(recognizerResult.getResultString());  
 **if** (!**""**.equals(**result**)) {  
 BaiduTranslationService.*getBaiduTranslationService*().request(**"en"**,  
 BaiduTranslationService.*getCode*(((MainActivity) getActivity()).getTargetLanguage()),  
 **result**,  
 **new** BaiduTranslationService.Response() {  
 @Override  
 **public void** response(String s, **int** status) {  
 **if**(status == ***STATUS\_OK***) {  
 **result** = **result** + **"\n"**+ s;  
 } **else** {  
 **result** = **result** + **"\n"** + **"Translation Services Unavailable!"**;  
 }  
  
 getActivity().runOnUiThread(()->{  
 Msg msg = **new** Msg(**result**, Msg.***TYPE\_RECEIVED***, ((MainActivity)getActivity()).getTargetLanguage(), s);  
 **msgList**.add(msg);  
 **adapter**.notifyItemInserted(**msgList**.size() - 1); *// 当有新消息时，刷新RecyclerView中的显示* **msgRecyclerView**.scrollToPosition(**msgList**.size() - 1); *// 将RecyclerView定位到最后一行* });  
 }  
 });  
  
 }  
 }  
}

result为系统识别的结果，若结果不为空，则把结果加入消息类Msg中，输出结果。

2.RecycleView气泡聊天界面：首先需要编写消息类Msg和适配器类MsgAdapter，那么是如何实现选择只输出左边或右边的气泡的呢，在适配器中的onBindViewHoler函数中编写如下代码即可。

**public void** onBindViewHolder(ViewHolder holder, **int** position) {  
 Msg msg = **mMsgList**.get(position);  
 **if** (msg.getType() == Msg.***TYPE\_RECEIVED***) {  
 *// 如果是收到的消息，则显示左边的消息布局，将右边的消息布局隐藏* holder.**leftLayout**.setVisibility(View.***VISIBLE***);  
 holder.**rightLayout**.setVisibility(View.***GONE***);  
 holder.**leftMsg**.setText(msg.getContent());  
 } **else if**(msg.getType() == Msg.***TYPE\_SENT***) {  
 *// 如果是发出的消息，则显示右边的消息布局，将左边的消息布局隐藏* holder.**rightLayout**.setVisibility(View.***VISIBLE***);  
 holder.**leftLayout**.setVisibility(View.***GONE***);  
 holder.**rightMsg**.setText(msg.getContent());  
 }  
}

3.翻译功能：拿英文翻译成目标语言举例，翻译功能也在上面介绍语言转文字功能是的onResult函数中。

**if** (!**""**.equals(**result**)) {  
 BaiduTranslationService.*getBaiduTranslationService*().request(**"en"**,  
 BaiduTranslationService.*getCode*(((MainActivity) getActivity()).getTargetLanguage()),  
 **result**,  
 **new** BaiduTranslationService.Response() {  
 @Override  
 **public void** response(String s, **int** status) {  
 **if**(status == ***STATUS\_OK***) {  
 **result** = **result** + **"\n"**+ s;  
 } **else** {  
 **result** = **result** + **"\n"** + **"Translation Services Unavailable!"**;  
 }  
  
 getActivity().runOnUiThread(()->{  
 Msg msg = **new** Msg(**result**, Msg.***TYPE\_RECEIVED***, ((MainActivity)getActivity()).getTargetLanguage(), s);  
 **msgList**.add(msg);  
 **adapter**.notifyItemInserted(**msgList**.size() - 1); *// 当有新消息时，刷新RecyclerView中的显示* **msgRecyclerView**.scrollToPosition(**msgList**.size() - 1); *// 将RecyclerView定位到最后一行* });  
 }  
 getTargetLanguage()用来根据上面的复选框来确定目标语言。s为翻译结果，s和result一起赋给新的result后输出。注意这里的翻译和输出是异步调用，记得在主线程中输出结果。

4.语音播报功能：语音播报需要设置监听器，点击按钮的时候触发语音播报，系统获取目标语言的文字并将其播报。监听器如下：

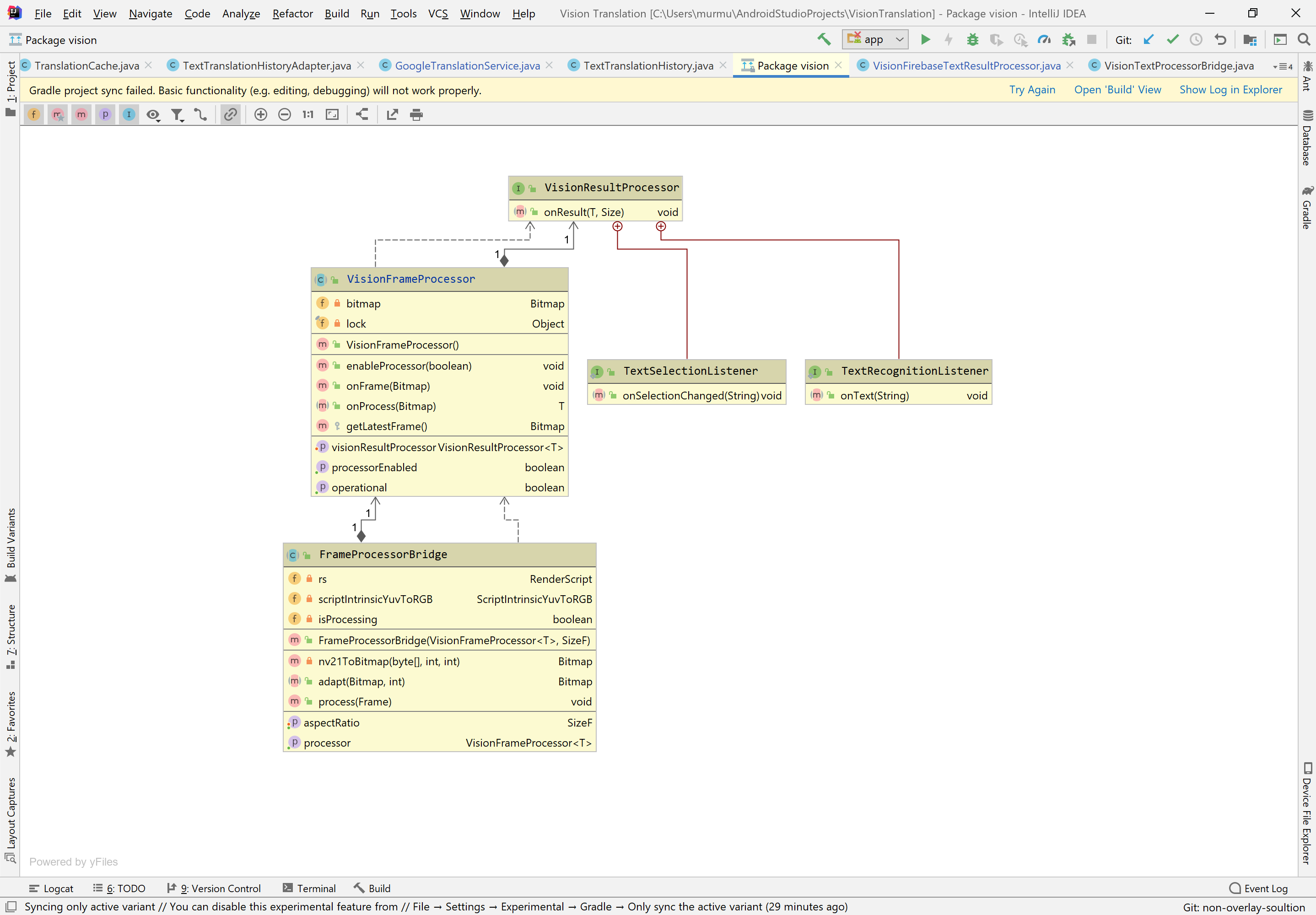
holder.**msgView**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **int** position = holder.getAdapterPosition();  
 Msg msg = **mMsgList**.get(position);  
  
 *//Toast.makeText(view.getContext(), "you clicked button!",Toast.LENGTH\_SHORT).show();* speak(  
 msg.getTarget(),  
 Helper.*getLocaleByLanguage*(  
 msg.getLanguage()),  
 view.getContext()  
 );  
 }  
});

当点击子项时系统会通过我添加在Msg中的getTarget()函数获取在翻译是存放在Msg中的目标语言并调用speak函数输出语语言播报。翻译板块和语音播报板块在相应的板块介绍，这里不多说。

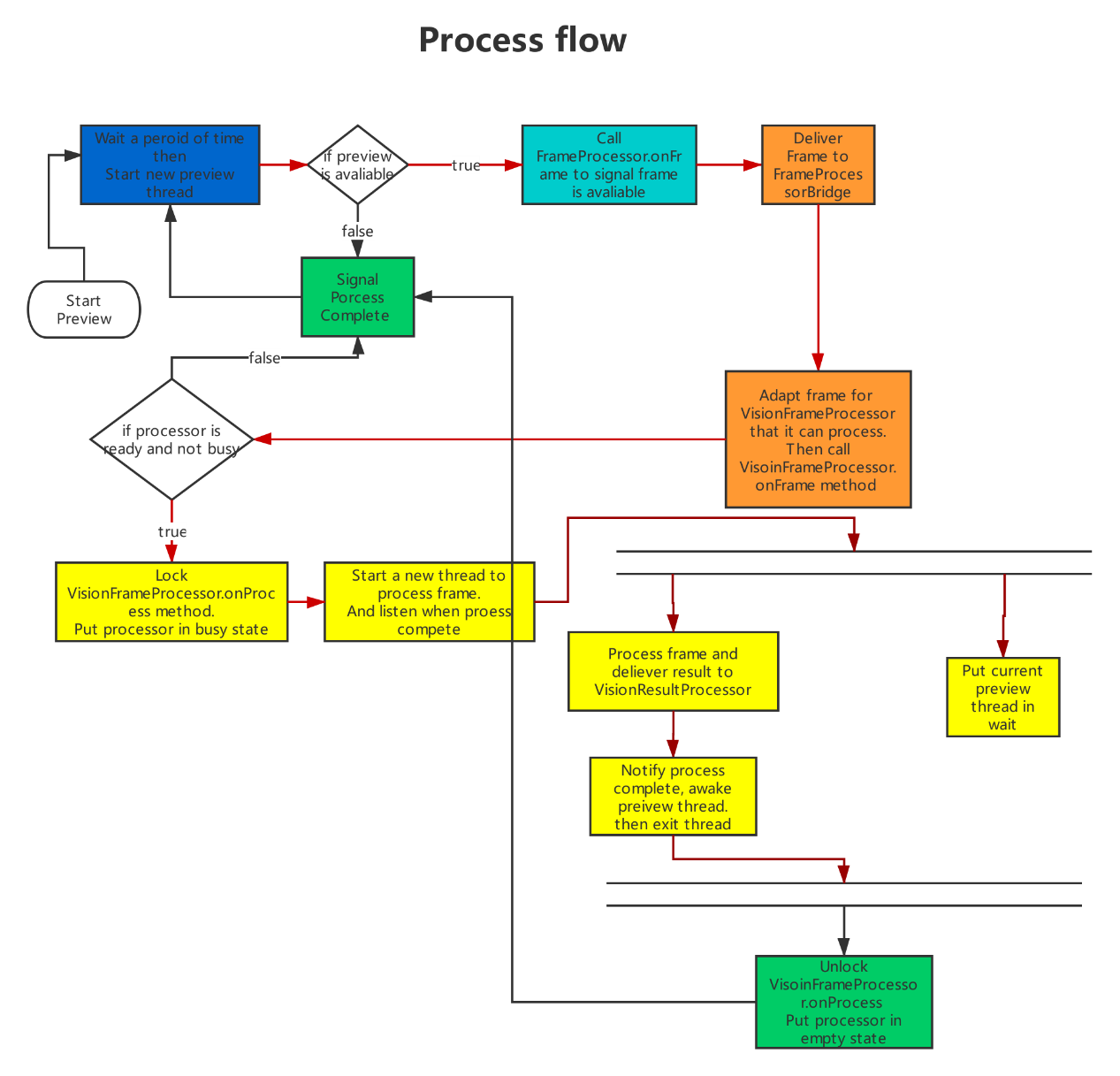
**（四）拍照翻译模块**

相机翻译由：帧处理器，结果处理器，结果追踪器，相机预览，相机管理，图形叠加六个部分组成。帧处理器负责处理图片并进行文字识别；结果处理器负责将帧处理器的处理结果呈现出来；结果追踪器负责追踪已经识别的文本，以加快识别速度；相机预览负责将相机预览显示到手机屏幕上，并向帧处理器输送预览；相机管理负责相机的开关，对焦以及帧处理器的管理；图形叠加是将一些图形叠加到Android的View之上，产生叠加效果，通常被称为：Graphics Overlay。

按照OO设计原则，上述几个组件都进行了不同程度的抽象和封装，实现对扩展开发，对修改关闭。帧处理模块、结果处理模块、和帧处理桥共同定义了一套关于图像处理的接口。其依赖关系如下。



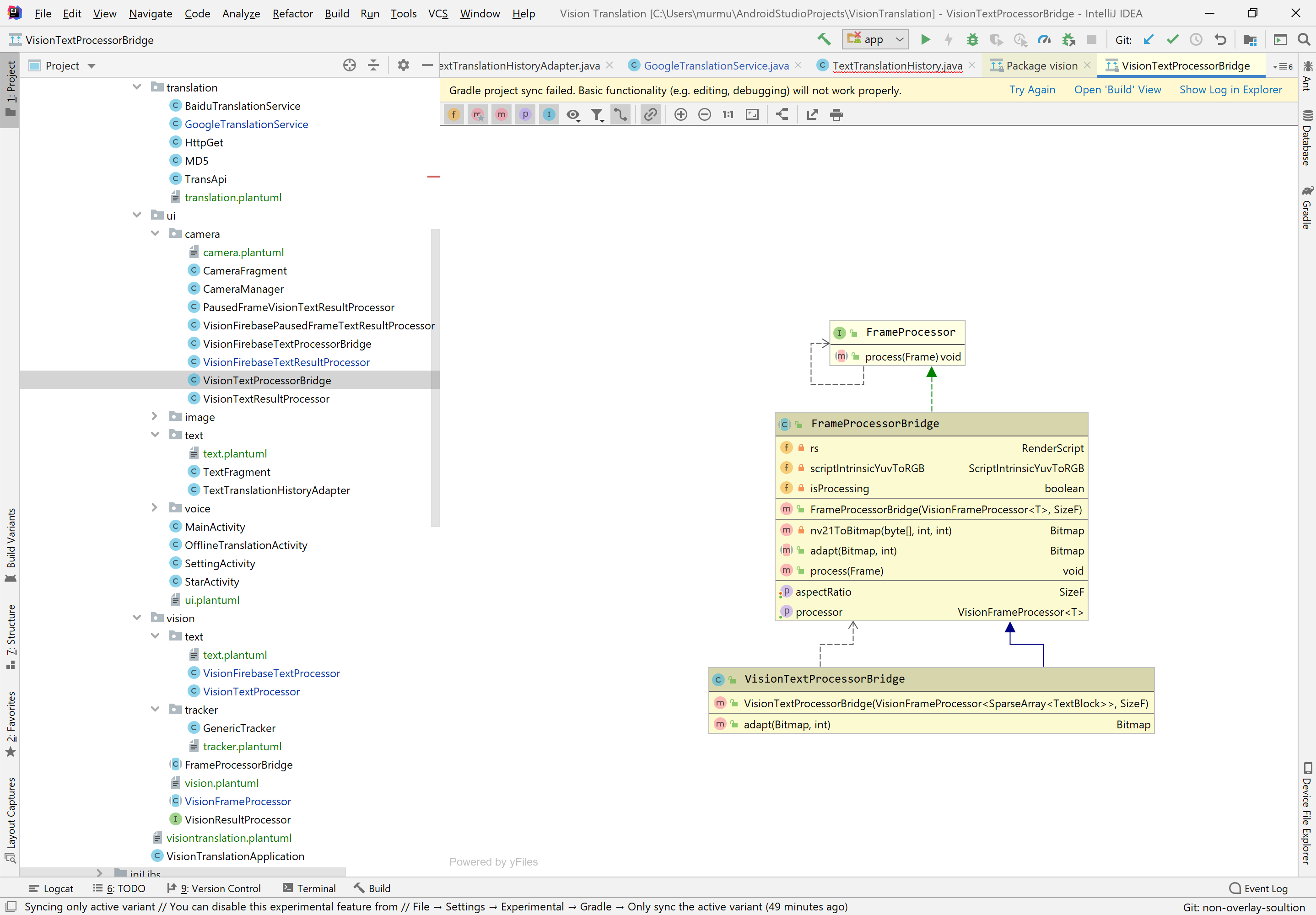
VisionFrameProcessor 所有处理器的抽象父类。它定义一一系列的方法规定了处理器该如何接受数据和传递结果，并加入一些信号控制，如关闭处理器，获取当前的处理帧，获取处理器是否运作正常。加入的处理器并发控制。VisionResultProcessor是所有结果处理器的都要实现的接口，规定了提交结果的内容。FrameProcessorBridge作为一个适配器，负责图像数据的适配工作，将各种格式的图片统一转换为处理器能够处理的数据。各部分处理的流程如下。



处理的开始是相机打开预览流。通过VisionFrameProcessorBridge将预览与帧处理器连接起来，对预览进行预处理，预处理完成后，将预处理的结果交给帧处理器进行文本识别，期间忽略其它帧，处理完成后将结果递交到结果处理器来处理结果，之后准备下一次处理。

**（1）预处理**

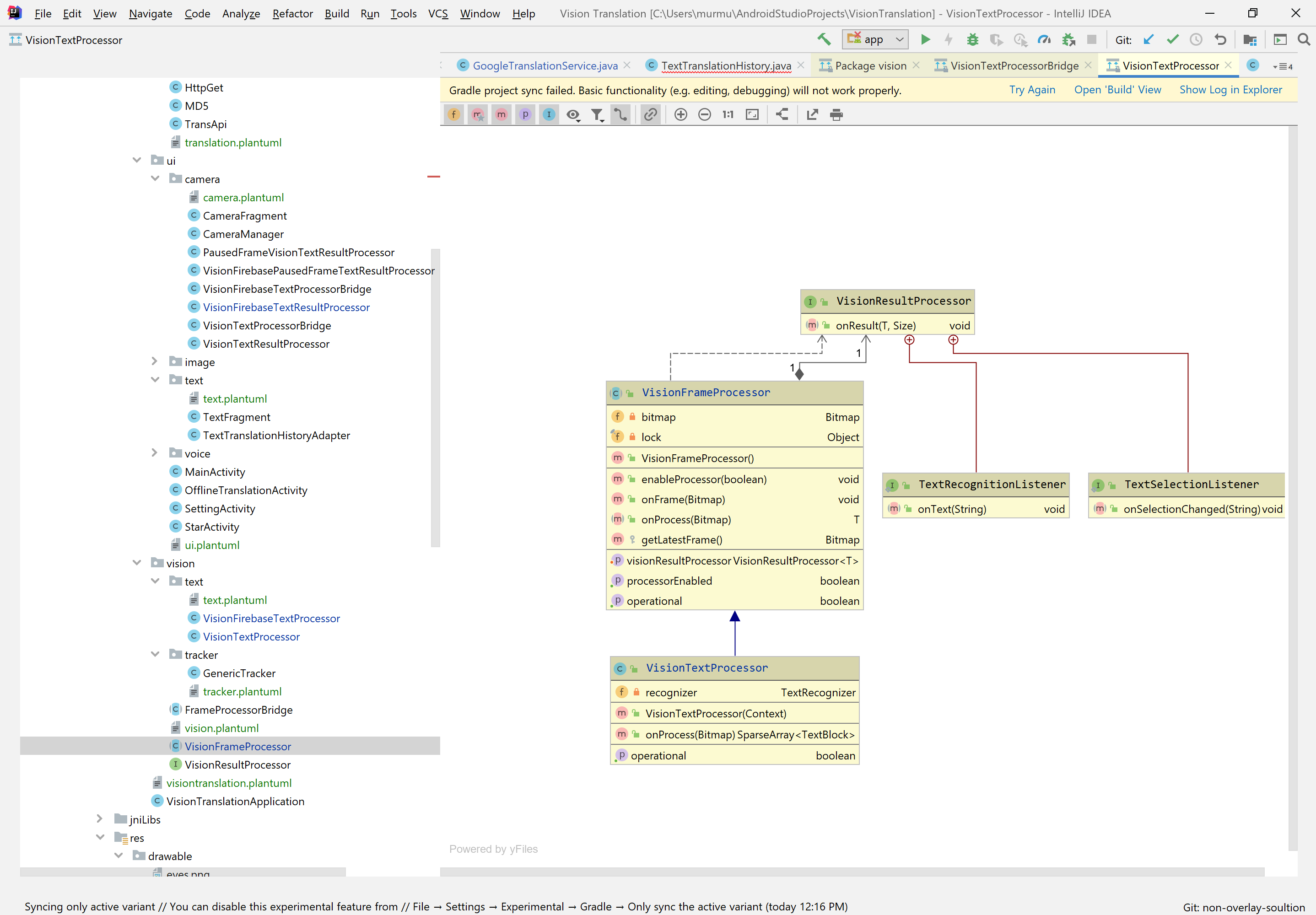
预处理主要工作是将，得到的相机预览原始数据转换为处理器可以处理的数据，并对预览进行旋转、裁剪使其大小符合显示要求。预处理的抽象类是FrameProcessorBridge。



以其中的一个实现类VisionTextProcessorBridge为例，在处理时通过process(Frame frame) void得到原始预览数据，之后用绘制脚本将原始的字节转换成Bitmap即调用nv21ToBitmap(byte[], int, int)。此时的到的图片并不是我们在手机屏幕上看到一样，因此需要对图像进行裁剪，旋转达到所见即所得的效果，方便处理结果。然后我们将图片交给帧处理器。

**（2）文字识别**

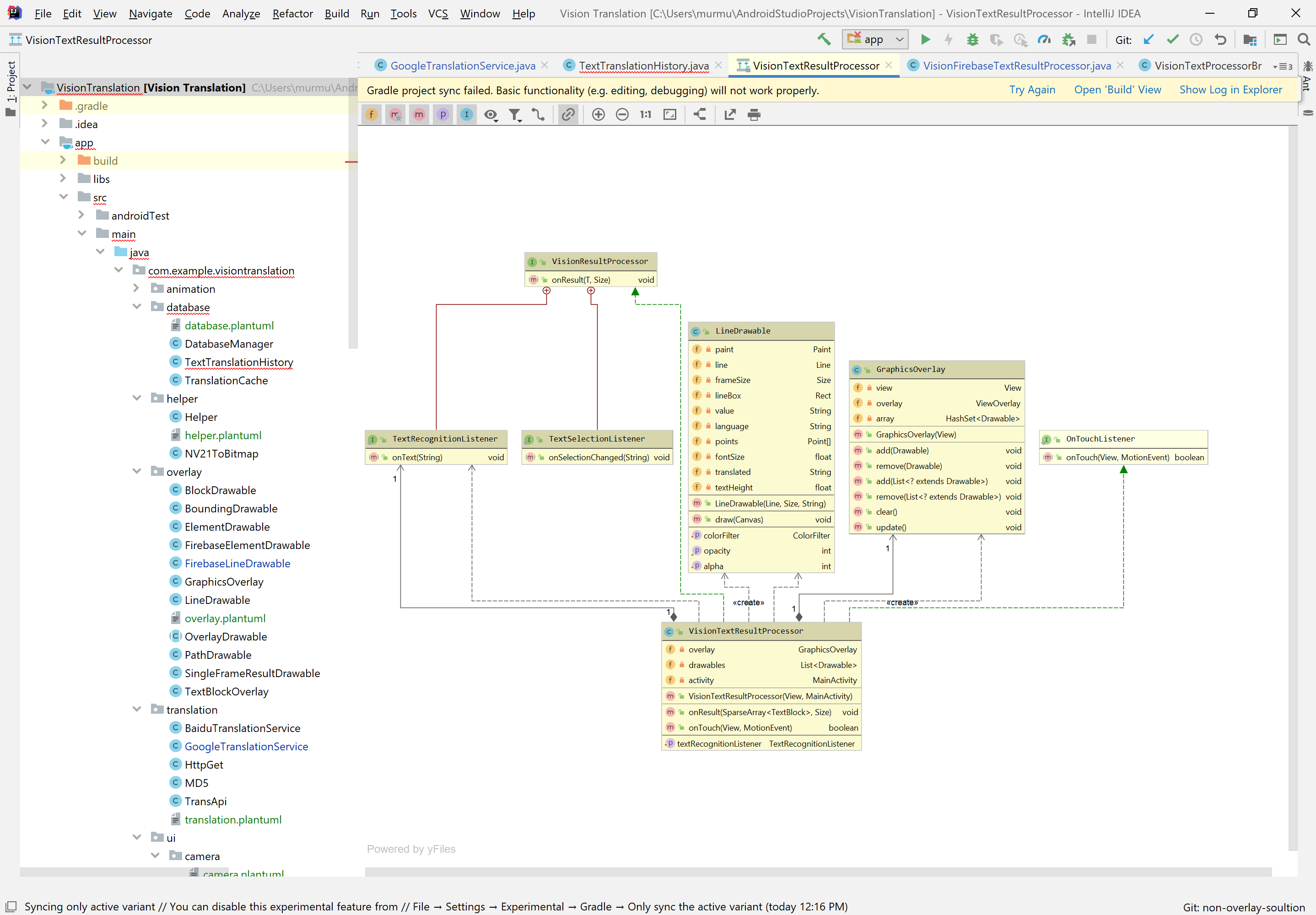
文字识别处理器继承自VisionFrameProcessor，在处理时为保障同步关系，例如结果处理必须在当前帧处理后，下一帧处理完成前结束，加入了同步和锁。VisionFrameProcessor的一个实现是VisionTextProcessor。该类包装了Google Mobile Vision的文字识别API。



**（3）结果处理**

结果处理的是显示处理结果到用户界面上，按照需求的不同有不同的解决方案。目前有两个实现类分别为VisionTextResultProcessor 和 PausedFrameTextResultProcessor，使用的场景各不相同。前者是在实时的将结果叠加到相机预览View之上，后者则是固定帧，将文本框位置显示出来，让用户选择文本。

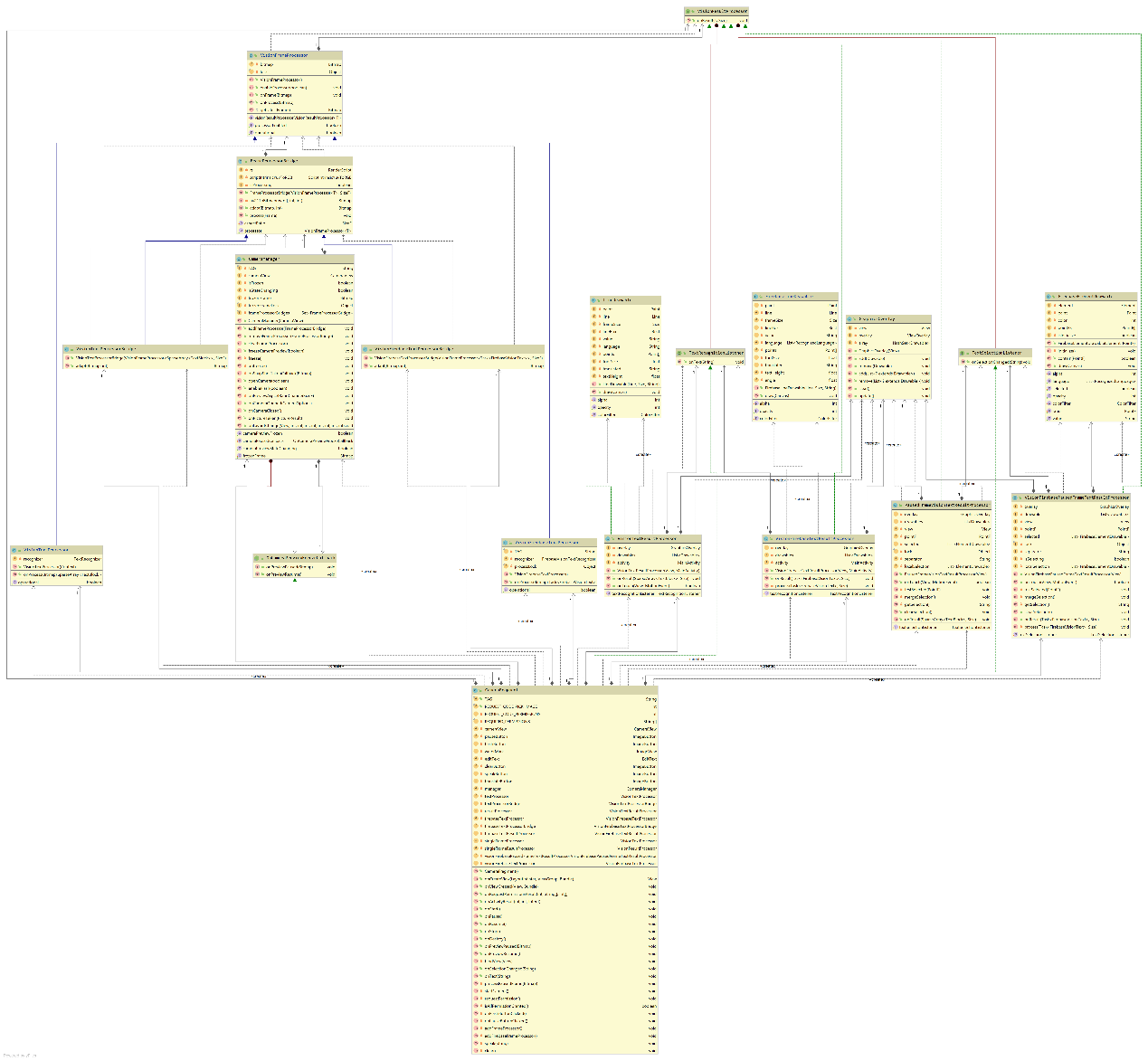
VisionTextResultProcessor的依赖如下



结果处理涉及的组件繁多，限于篇幅，不在此处赘述。

**（4）相机管理**

相机管理的职责是管理，即管理处理器的添加删除，也管理处理器的关闭开启，同时负责相机预览暂停的一些协作。结构复杂，这里放上类图。



**五、成果展示**

**（一）APP运行主界面文本翻译模块**

界面上方可切换源语言和目标语言

输入文字，点击文本框右下角图案开始翻译

点击左上角音量图标进行语言播放



**（二）拍照翻译模块**

相机的左下部位按钮可打开手机手电筒功能

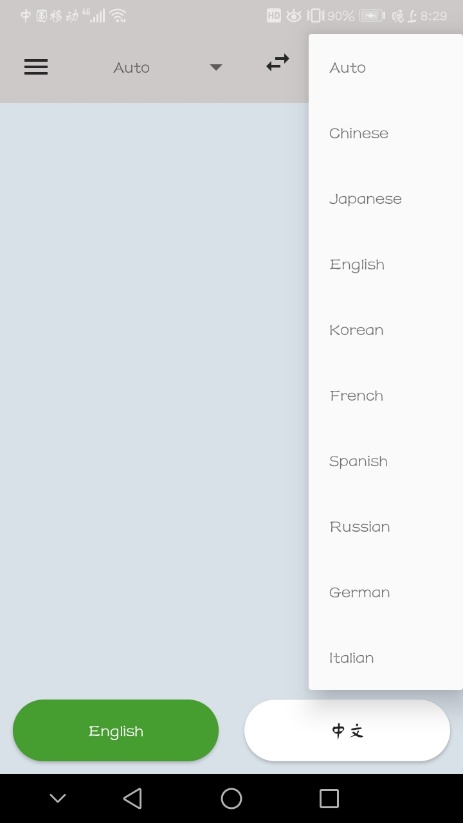
相机的的右下部位按钮可以是画面暂停





**（三）语音翻译模块**

选择目标语言



长按按钮录入语言并翻译

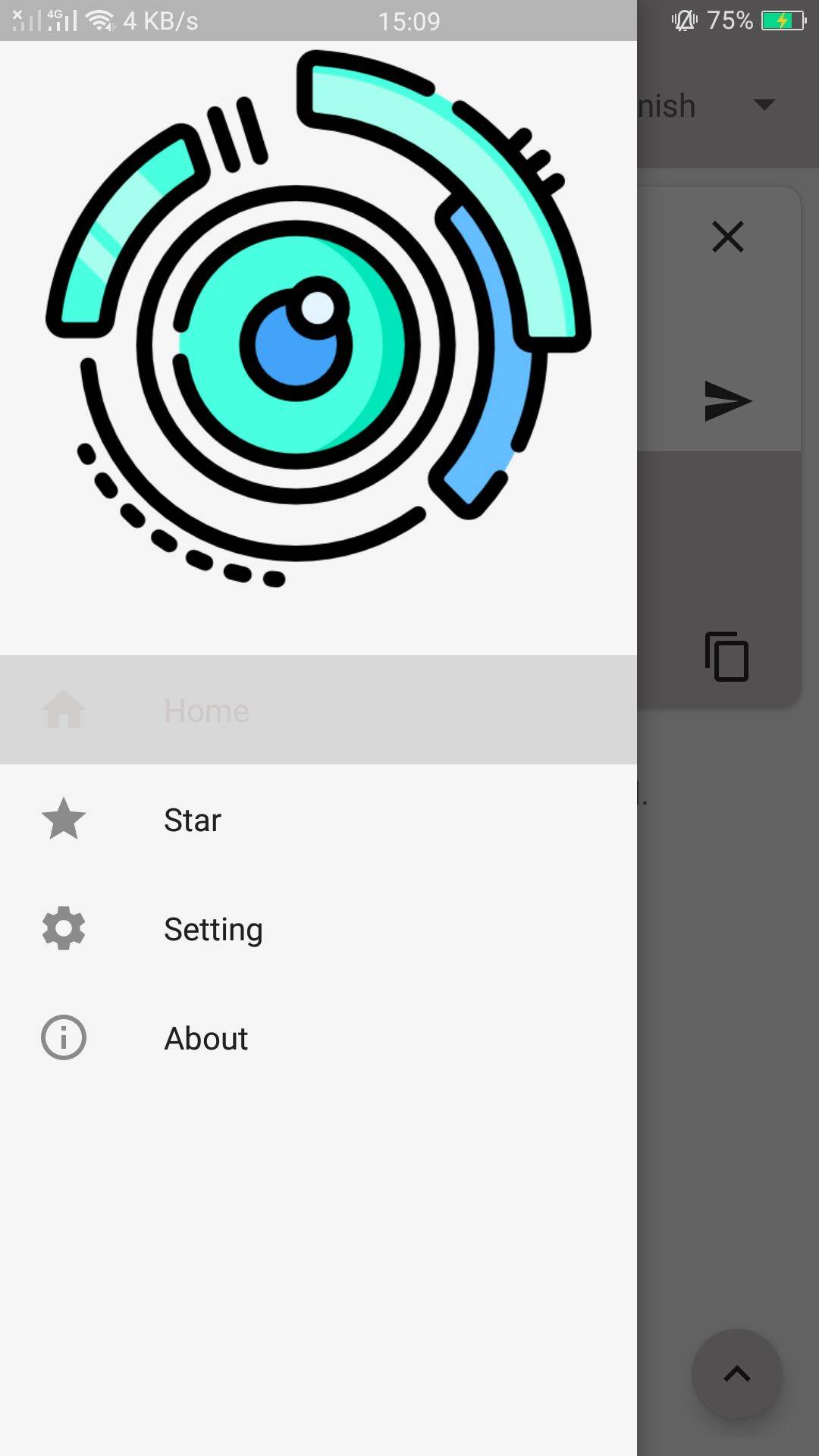


点击子项播报目标语言



**（四）菜单模块**

star查看收藏的文本翻译，setting设置，about关于产品的信息



**六、总结**

**（一）项目成果**

项目最终完成了一个集文本翻译，相机翻译，语言识别翻译，语音播放于一体Android终端的app。

**（二）小组分工**

赵文：App主界面，应用程序框架，Google离线文本翻译，翻译缓存，翻译历史记录，相机翻译，菜单。

付康：语音识别，文字气泡，调用文本翻译接口和tts接口，ppt，视频。

夏谦：百度在线文本翻译，tts文本转语音。

**（三）收获**

这是我们第一次做Android App开发项目，历时将近2个月，过程中遇到了很多困难，也学到了很多知识，从入门到开发项目。

我们从Android 程序最基础的活动学起，一步一步的往前走，对活动的运行机制有了一定的理解，学习如何在活动界面中布局。在开发项目期间，我们的编码能力都有很明显的提升，对Java语言的使用更加熟练，学会使用多线程和为布局添加各种触发事件，同时锻炼我们上网查阅资料和自主学习的能力，学习使用第三方API和系统库。

最后将小组各成员的程序代码整合时，出现了各种bug，最后都被我们一一解决，锻炼了我们与人交流沟通的能力，培养了团结协作的优秀品质。