2017 年夏季 Java 小学期大作业

实验报告

陈经基　　高信龙一　　张蔚

2015011358　　2015080060　　2015011352

2017 年 9 月 14 日

目录

1 功能实现 3

2 小组分工 4

3 具体实现 5

3.1 models 6

3.1.1 models 与其他模块的接点：NewsManager 类 6

3.1.2 INewsIntroduction 接口 8

3.1.3 INewsList 接口 9

3.1.4 INewsDetail 接口 9

3.1.5 获取最新新闻功能：新闻的在线获取和离线缓存 10

3.1.6 获取新闻详情功能 10

3.1.7 新闻收藏功能 11

3.1.8 语言朗读新闻功能 11

3.1.9 新闻搜索功能 11

3.1.10 新闻 SDK 分享至社交 APP 功能 12

3.1.11 新闻推荐功能 12

3.1.12 网络操作 12
3.1.13 数据库操作 13
3.1.14 单元测试 13
3.2 Contract 14
3.2.1 View 14
3.2.2 Presenter 14
3.3 主界面 14
3.4 分类列表 16
3.5 新闻 17
3.5.1 新闻列表 17
3.5.2 新闻详情 18
3.6 收藏 19
3.7 搜索 19
3.8 设置 20

4 总结与心得 21

1 功能实现

在本次实验中所实现的基础功能以及加分功能分别如表 1、表 2所示；

功能	子功能	百分比	是否实现
系统支持	要保证程序在安卓机上正常运行，测试过程中程序不崩溃	5	是
页面布局	布局合理，点击处理正确	10	是
分类列表	删除和添加操作	10	是
新闻列表	正确显示新闻列表的消息，布局和展示，点击进入新闻详情页面正确	10	是
	实现新闻的本地存储，看过的新闻列表在离线的情况下也可以浏览	10	是
	上拉获取更多新闻	5	是
	新闻是否看过的页面灰色标记	5	是
	新闻搜索	5	是
分享功能	使用微信、微博等 SDK 分享，新闻详情页面点击分享可以分享到常用的 app，分享内容带有新闻摘要、URL 和图片	10	是
	新闻详情页面点击收藏的添加和删除，实现收藏新闻的本地存储。收藏也的正确展示，点击可以进入新闻详情等	10	是

表 1: 基础功能实现说明表格

功能	子功能	百分比	是否实现
新闻推荐	根据用户看过的新闻推荐相关的新闻，参考今日头条等	10	是
语音播报	可以语音读出新闻等	20	是
新闻链接	用户可以跳转到相应的百科词条等	5	是
用户体验	新闻屏蔽功能，通过进一步询问用户想要屏蔽掉关于什么内容的新闻而实现基于关键词的屏蔽等	5	
	夜间模式，用户可以调整背景色等	5	是
	文字模式和图片模式转换，文字模式不显示图片，帮助用户节省流量	5	是
	流畅性强	-	是
	界面颜值高	-	
	根据新闻文本补上相关的图片	-	
代码和项目管理	使用了较好的框架	-	是
	有较完整的单元测试	5	是
	使用了 github 等好用的代码版本管理工具	2	是[a]
	maven 或者 gradle 等项目管理工具	2	是
	其他提高编码效率的工具或操作	-	

表 2: 加分功能实现说明表格

[a]https://github.com/AmadeusChan/AndroidAppProject

2 小组分工

功能	实现者
底层接口，具体包括有： 最新新闻列表、新闻搜索、新闻推荐、 新闻收藏、新闻详情的数据获取、 标记已读新闻、语音播报、新闻分享	陈经基
分类列表的获取和布局	张蔚
新闻详细页面布局	高信龙一
信息本地存储	陈经基
更多新闻获取	张蔚
已读/未读新闻记录	张蔚
新闻搜索功能	张蔚，高信龙一
分享到微信和微博	陈经基，张蔚
新闻详情收藏	张蔚，高信龙一
新闻推荐系统	陈经基，张蔚
语音播报	陈经基
关键词百科词条链接	陈经基，张蔚
夜间模式	张蔚
文字模式和图片模式	高信龙一

表 3: 小组分工说明表格

3 具体实现

整体框架采用 MVP 模式。M 对应 Model，负责数据的存储、检索、操作。V 对应 View，在本安卓软件中即为各个 Activity，负责 UI 元素的初始化，建立 UI 元素与 Presenter 之间的关联，同时自己处理一些简单的逻辑操作。P 对应 Presenter，由于 Activity 类的职责不断增多，以致于变得臃肿庞大，通过将一些复杂的逻辑处理分离到一个别的新类中，让 View 更加专注于处理数据的可视化以及用户的交互，同时让 Model 只负责数据的处理。此外，由于 Presenter 是通过 Interface 与 View(Activity) 进行交互的，我们可以通过自定义类实现这个 Interface，来模拟 Activity 的行为对 Presenter 进行单元测试，省去了大量的部署及测试的时间。

归纳而言，这样的设计模式有以下几个优点:

- 降低耦合度
- 模块职责划分明显
- 利于测试驱动开发
- 代码复用
- 隐藏数据
- 代码灵活性

在本项目底下按照具体功能分别分为 channel, favorite, main, models, news, search, settings 七个子包，接下来的内容将逐个介绍这些子包的具体功能以及其实现方法；

此外，本项目的包名为 com.java.a31.androidappproject，与实验要求中的 com.java. 组号格式略有不同，这是因为 java 中包的命名是不允许数字开头的，即 com.java.31 这种包名是不合法的，因此对包名做了些许的改动，并且这种改动不会影响到包名的唯一性；

3.1 models

models 模块负责提供底层数据的获取和处理等操作，对应于 MVP 设计模式中的 Model 部分，其目的是将项目的其他部分与网络、数据库等操作完全隔离开来，从而有效地降低代码耦合程度，提高分工效率；下文将分别对本模块中重要的类、接口以及功能进行详细介绍；

3.1.1 models 与其他模块的接点：NewsManager 类

该类是项目其他部分使用 models 模块的所用功能的入口；为了防止实例化多个 NewsManager 对象从而导致多个不同对象操作 models 里的数据产生混乱，该类被设计为 Singleton 类，调用其 getInstance() 实例可以获取到其全局唯一一个实例，并且第一次调用需要传入一个 Context 对象进行初始化；

在本项目中几乎所有的涉及到数据操作（即与 UI 无关）的功能都在 NewsManager 类中可以找到相应的进行底层操作的方法，下文中将对这些方法进行列举并且简要描述其功能；

- void getInstance(), void getInstance(Context context)：这两个方法用于获取一个 NewsManager 实例；
- INewsList getLatestNews(int mode)：获取新闻列表，返回的是一个实现了 INewsList 接口的对象，其代表一个新闻列表；mode 参数表示对文本模式或者是图片模式的选择；

- void getNewsDetails(String ID, int mode, INewsListner<INewsDetail> listener)：在 mode 模式下获取 ID 相对应的新闻的详细信息，获取到的详细信息是一个 INews-Detail 接口的实例，并且考虑到该功能涉及到网络操作，为了防止 UI 线程（主线程）的卡顿，该功能的实现代码需要异步执行，因此采用了类似观察者模式的方法，给当前方法传入了一个 listener（其类型为 INewsListner 接口）进行监听，一旦网络操作完成，则会调用传入 listner 对象的 getResult(INewsDetail result) 方法，将获得的详细信息交给其进行处理；
- void setAsFavorite(INewsDetail newsDetail)：收藏传入的新闻；
- void setAsNotFavorite(INewsDetail newsDetail)：取消对该新闻的收藏；
- INewsList getFavoriteNews()：获取被收藏的新闻列表；
- boolean isFavoriteNews(String ID)：判断 ID 对应的新闻是否被收藏；
- void speakText(String text)：调用在线的 TTS API 读出传入文本；
- void stopSpeaking()：停止调用上一个方法的语音朗读；
- List<String> getCategoryList()：获取当前 APP 需要显示的新闻分类；
- void addCategory(String category)：增加当前 APP 需要显示的新闻分类；
- void deleteCategory(String category)：在当前 APP 需要显示的新闻分类列表中删除某一个分类；
- INewsList searchNews(String keyWord, int mode)：按照 keyWord 作为关键词进行搜索，mode 表示是否为图片模式；
- void jump2Baike(String keyWord)：跳转到 keyWord 所对应的百科页面，该方法在最终的实现中没有被采用，而是使用了直接在新闻详细页面的文本中添加超链接的方法；
- share2Weibo(Activity activity, String url, String introduction, String image)：在传入的 activity 中发起分享到微博操作，分享内容包括传入的网址，简介以及图片（的 url）；
- share2Wechat(String url, String introduction, String image, int mode)：发起分享到微信操作，分享内容与上一个方法类似，mode 用于区分分享到朋友圈和分享给朋友两者模式；
- INewsList getRecommendNewsList(int mode)：获取推荐新闻列表；

- boolean addReadNews(String ID)：将某条新闻设置为已读；
- boolean isReadNews(String ID)：判断某条新闻是否已读；

如上所示，NewsManager 类提供了了新闻列表的获取、删除、添加、获取详细信息、收藏以及取消、语言朗读新闻、新闻搜索、SDK 分享、新闻推荐、已读新闻的标记等一系列功能的底层支持，并且完全地将这些功能中的繁琐细节隐藏起来，比如说在下拉获取更多新闻列表的时候，models 模块会自动根据网络状态来判断是提供在线 API 获得新闻，还是从数据库中访问缓存下来的新闻列表等，有效地方便了其他模块的实现，这些 models 部分隐藏起来的细节将在后文中一一进行介绍；

在上文中提及到了 INewsList、INewsDetail、INewsIntroduction 三个负责具体数据表示的接口，这三个接口用于规定新闻 APP 中对应的新闻列表、新闻详情、新闻列表中的每条新闻的简介这三种元素中所包含的数据所对应的类应当具有的方法，这些方法与这些元素的特性相关；使用接口作为 NewsManager 中与获取新闻列表等这些操作相关的方法的返回类型，将这些数据类的具体实现向其他模块隐藏了起来，其优点是无论这些数据类的具体实现如果改变，都不会对 UI 的代码造成任何影响；事实上，在开发过程中，返回最新新闻列表相关的具体的类进行过数次变更，然而项目的负责其他模块的合作者完全不需要因此增加任何工作负担；下文中将对这三个接口进行介绍；

3.1.2 INewsIntroduction 接口

该接口规定了表示新闻简介（即在新闻列表中的一项）的类需要拥有如下方法：

- String getClassTag(); 获取新闻类别；
- String getID(); 获取新闻 ID；
- String getSource(); 获取新闻来源；
- String getTitle(); 获取新闻标题；
- String getTime(); 获取新闻时间；
- String getURL(); 获取新闻对应网页的 URL；
- String getAuthor(); 获取新闻作者；
- String getLanguage(); 获取新闻语言信息；
- String getVideos(); 获取新闻包含视频；

- String getIntroduction(); 获取新闻简介；
- List<String> getImages(); 获取新闻包含图片；
- boolean isRead(); 判断新闻是否已读；
- boolean isFavorite(); 判断新闻是否被收藏；

3.1.3 INewsList 接口

该接口规定了给 APP 中的新闻列表这一元素提供数据的类所需要实现的方法；而 APP 中的新列表最主要的操作就只有两种，第一种是下拉刷新列表，这就对应于接口中的 void reset() 方法，表示刷新新闻列表中的数据；而第二种是上拉获取更多内容，这就对应了该接口中的 getMore() 方法，考虑到对新闻分类列表的支持，总共提供了三种 getMore 方法，如下所示：

- void getMore(int size, int pageNo, INewsListener<List<INewsIntroduction» listener); 表示获取规定了页面大小的第若干页新闻；
- void getMore(int size, int pageNo, int category, final INewsListener<List<INewsIntroduction» listener); 表示获取规定了页面大小与新闻种类的第若干页新闻；
- void getMore(int size, INewsListener<List<INewsIntroduction» listener); 表示获取更多的若干条新闻；

getMore() 方法中都需要提供一个 listener 作为参数，这是因为在这个方法中涉及到了网络操作，为了防止卡顿主线程，因此这些操作都是异步执行的，因此最后需要调用 listner 的 getResult 方法来传回最终结果；

其次，为了给实现新闻屏蔽功能实现支持，INewsList 还规定了 setFilter(INewsFilter filter) 方法，在调用 getMore 方法的时候，会根据 filter 的判断，屏蔽掉需要屏蔽的新闻；

INewsList 接口的另一个优点是其将收藏新闻列表，最新新闻列表，搜索结果的新闻列表，缓存的新闻列表的表示都统一起来，对于外部模块，这些列表都是一个 INewsList 实例，方便 UI 部分复用显示新闻列表的代码；

3.1.4 INewsDetail 接口

该接口规定了给 APP 中新闻详细页面提供数据的类应当具有的方法；这个接口继承了 INewsIntroduction 接口，并且新增了如下方法：

- List<String> getPersons(); 获取新闻中相关的人物列表；
- List<String> getLocations(); 获取新闻相关的地点；
- List<String> getKeyWords(); 获取新闻关键词；
- String getContent(); 获取新闻详细内容；

3.1.5 获取最新新闻功能：新闻的在线获取和离线缓存

外部模块获取最新新闻的列表所需要使用的方法是 NewsManager.getLatestNews(int mode)，该方法返回的 INewsList 对象是一个 GeneralNewsList 实例，该类负责提供最新的新闻列表数据，并且会根据当前的网络状况自动判断应当从网络还是数据库中获取数据；该类中封装了两个实现了 INewsList 接口的类的对象 onlineList 和 offlineList，前者是一个 NewsList 对象（该对象将在下文介绍网络操作的部分中进行介绍），负责从网络中获取新闻列表信息；而后者是一个 CachedNewsList 对象，负责从数据库中获取缓存下来的新闻列表；

每一次从网络中获取到新闻列表数据，都会自动将调用数据库操作相关类的方法，将这些数据缓存到数据库中，因此缓存新闻列表操作对外部模块是不可见的，也是不应该显式进行操作的；

3.1.6 获取新闻详情功能

外部模块获取某新闻的详细信息所需要调用的方法是 NewsManager.getNewsDetail(String ID, INewsListener listern)，该方法将调用 NewsDetailFetcherFromInternet 类中的静态方法 fetchDetail 从网络上获取详细的新闻数据，并且在网络操作完成之后将其传给 listner 进行处理；

考虑到每一次获取新闻详情数据的操作都意味着用户点开了某一个新闻详情页面，因此调用该方法会自动在数据库中将这条新闻标记为已读状态，之后任何表示该新闻的 INewsIntroduction 对象的 isRead() 方法都会返回 true；

另外一个 getNewsDetail 的功能是，在离线状态下，调用该方法获取一条已经收藏的新闻的详细信息，该方法会调用数据库相关类中的方法来获得缓存下来的新闻信息并返回，将在线离线的新闻详情获取都统一到同一个方法也是将外部模块与网络环境彻底隔离开来这一想法的体现；

3.1.7 新闻收藏功能

外部模块改变某条新闻的收藏状态以及获取收藏的新闻列表的时候，需要调用的方法是 NewsManager 中的 setAsFavorite, setAsNotFavorite, getFavoriteNews，前两个方法分别会将某条新闻设置为收藏获取取消收藏，并且收藏新闻的方法还会将新闻的详情信息缓存到数据库中，保证其在离线状态下也可以通过上文提及的 getNewsDetail 方法来访问得到；而 getFavoriteNews 将返回收藏的新闻列表，其返回类型是 INewsList，因此对于外部模块，处理收藏新闻列表与处理其他新闻列表没有任何区别，方便代码复用；

getFavoriteNews 返回的 INewsList 对象是一个 FavoriteNewsList 类（实现了 INewsList 接口）的实例，在调用了该方法之后，NewsManager 就会调用数据库相关的类，获取已经收藏起来的新闻列表信息，然后生成一个 FavoriteNewsList 的实例返回；

3.1.8 语言朗读新闻功能

外部模块调用这部分功能所使用的方法为 NewsManager 中的 speakText, stopSpeaking 两个方法，前者的作用是访问在线的 API 然后进行后台的语音播报，而后者则会将后台的语言播报停止下来，提供后者这个方法这种设计是合理的，因为如果在当前用户退出新闻详细页面甚至打开一个新闻语言播报的时候，原来的语音播报仍然在后台运行，这会造成糟糕的用户体验，并且为了防止两个语音播报同时进行，在 speakText 中默认会先调用一次 stopSpeaking 方法；

TTS（Text to speech）功能所使用的在线 API 由 Voice RSS[1]网站提供，其接受使用 HTTP GET 方法将需要语言播放的文本上传，并且在 response 中返回一个 MP3 文件；考虑到这部分功能同样涉及到网络操作，因此不可能在主线程中实现，因此使用了 AsynTask 类进行异步实现；

3.1.9 新闻搜索功能

新闻搜索功能对应于 NewsManager.searchNews 方法，其返回类型与获取收藏新闻列表、获取最新新闻一样，都是 INewsList；

新闻搜索功能和获取在线的新闻列表功能部分非常相似，都是使用 HTTP GET 方法将某些参数传给 RestFul API，然后获取一个 Json 格式的新闻列表，因此可以将这两个功能交给同一个类负责，这个类就是 NewsList 类，也是上文中获取最新新闻部分用于获取在线新闻的类，对于该类将在后文的网络操作部分进行详细介绍；

[1]http://www.voicerss.org/api/documentation.aspx

3.1.10 新闻 SDK 分享至社交 APP 功能

分享功能对应到 NewsManager 中的 share2Weibo, share2Wechat 两种方法，分别对应分享到微博和分享到微信两种分享方法，并且分享到微信又分为分享到朋友圈和好友两种模式, 由 share2Wechat 中的 mode 参数决定；

总体而言，使用 SDK 分享到社交 APP 一般包括了如下几个步骤:

(1) 在对应社交 APP 的开发者网站上注册成为开发者，并且给自己开发的移动应用进行注册，获取相应的 APPID 和 APPSECRET（某些网站可能需要对应用进行审核之后才会发放 APPID 和 APPSECRET，比如微信）；

(2) 获取这些社交 APP 提供的 SDK 工具，有些 SDK 工具会直接通过 jcenter 进行方法，比如说微信和 facebook 的 SDK 工具，而有些则需要手动下载 jar 包并且添加到当前项目中，比如说微博；

(3) 参考社交 APP 的开发者平台上提供的样例，实现分享操作；

在实现这部分功能的过程中的比较麻烦的地方在于，微信的第三方应用审核在 9.13 号才批准下来，这使得我们不得不在项目基本完成的两三天之后还要仓促地进行新功能的添加；

3.1.11 新闻推荐功能

推荐功能对应到 NewsManager 中的 getRecommendedNewsList 方法，返回类型仍然是 INewsList, 所实现的新闻推荐算法比较简单，仅仅是将阅读过的新闻的关键词记录下来，然后每次推荐的时候选出出现频率若干高的关键词对应的新闻进行推荐，这种算法有一个显而易见的缺点是如果用户只看了带有某一个新闻，就只会推荐与这个新闻高度相似的内容，无法满足用户尝试阅读其他未读过的类型的新闻的需要；

3.1.12 网络操作

接下来介绍模块中的网络操作部分，这部分操作由 NewsList 和 NewsDetailFetcher-FromInternet 这两个类实现，前者负责获取网络中的新闻列表，后者则负责获取新闻详细信息；

首先介绍 NewsList 类，该类中的 getMore 方法会从网络上的 API 获取新闻列表信息；在实现的过程中使用了 volley 包方便网络操作，该包中进行发送 HTTP REQUEST 和获取 response 的方法是异步的，因此不会对 UI 线程造成堵塞；获取到的 response 是 json 格式的，

因此会使用 android 自带的 JsonObject 等类进行 Json 解析，最后将获取的新闻列表存在一个 List<INewsIntroduction> 对象中传给 listner，让其实现 UI 中对获取到数据的响应；

其次介绍 NewsDetailFetcherFromInternet 类，同样该类使用了 volley 进行异步的网络操作，并且使用 JsonObject 进行 Json 的解析，最后将获取到的信息封装成一个 INewsDetail 对象，作为结果传给 listener；

3.1.13 数据库操作

新闻列表的缓存，新闻的收藏这些都需要使用数据库储存数据，models 中数据库操作相关的类是放在 database 子包中的，其中与外界进行交流的类为 MyDBHelper, 该类中按照安卓开发者网站上的提示[2]继承了 SQLiteOpenHelper 类以便于使用安卓自带的 sqlite 数据库进行数据库操作，这些操作包括了新闻列表的缓存，看过的新闻的标记，新闻分类列表的储存，新闻收藏列表和被收藏新闻的缓存，看过的新闻涉及到的关键词的储存（新闻推荐相关）的功能；

其中使用 sqlite 数据库的具体操作与一般的 sql 数据库的操作非常类似，因此在此不再赘述；

在数据库操作中遇到的一个比较麻烦的地方在于，在最开始设计获取缓存新闻列表的类（应当是 INewsList）的子类的时候，选择了一次性将数据库中的缓存新闻列表都读出来交给新闻列表对象的方法，这使得创建新闻列表对象变得非常慢，甚至会给主线程造成 1 2 秒时间的卡顿；后来优化的方法在于延迟了新闻列表数据真正在数据库中获取的时间，在创建负责缓存新闻列表的 INewsList 对象的时候，只会将数据库的 Cursor 传给它，之后具体的对数据库的方法则全部延迟到调用 getMore 方法的时候才完成；事实上在获取网络上的新闻列表部分使用的也是相似的方法，真正的对网络的访问也是发生在 getMore 方法里面的；

3.1.14 单元测试

值得一提的是在 models 模块大量使用了 android instrumentation test 的单元测试[3]，该种类的单元测试满足了在安卓环境下测试各个模块的要求，models 中的几乎每个方法都有对应的单元测试，这使得开发该模块的过程中不对 UI 模块的开发造成任何影响，极大地提高了开发效率；

[2]https://developer.android.com/training/basics/data-storage/databases.html?hl=zh-cn

[3]https://developer.android.com/training/testing/unit-testing/instrumented-unit-tests.html

3.2 Contract

我们仿照 Google 用于展示 MVP 架构的官方样例[4]，将 View 和 Presenter 接口单独放到一个 Contract 接口中。

3.2.1 View

应用中所有用户可见的 Activity 和 Fragment 等 UI 元素都实现了相应的 View 接口，对应 MVP 架构中的 View 部分。这些 View 接口都继承自 BaseView 接口，并根据需求增加了用于给 presenters 改变 UI 用的函数。

```
public interface BaseView<T> {
    void setPresenter(T presenter);
}
```

3.2.2 Presenter

应用中所有 View 都有对应的 presenter 与其进行交互，这些 presenter 对应 MVP 架构中的 Presenter 部分。所有 Presenter 接口都继承自 BasePresenter 接口，并根据需求增加了用于给 view 通知事件用的函数。

```
public interface BasePresenter {
    void start();
}
```

3.3 主界面

本模块负责本软件的主要活动 (Main Activity)。Main Activity 通过 MainActivity 类实现，大致分为以下三部分：

- 新闻片段 (newsfragment)；
- 收藏片段 (favouritefragment)；
- 设置片段 (settingsfragment)。

[4]https://github.com/googlesamples/android-architecture

因为 Fragment 有自己的生命周期和接收、处理用户的事件，通过将各个主要功能分散到设计的 Fragment 中，这样就避免在一个 Activity 里面写一堆事件、控件的代码，从而 Activity 不需要管理 View Hierarchy 的复杂变化。同时，Fragment 能帮助我们轻松地创建动态灵活的 UI 设计，可以适应于不同的屏幕尺寸，这样还可以动态的添加、替换、移除某个 Fragment。

MainActivity 和 MainPresenter 遵循 MainContract 给出的接口：

```
public interface MainContract {
    interface View extends BaseView<Presenter> {
        void switchToHome();
        void switchToFavorite();
        void switchToSettings();
    }

    interface Presenter extends BasePresenter {
        boolean switchNavigation(int id);
    }
}
```

总体上，通过调用 MainPresenter 类来实现各个片段之间的切换，具体而言，其界面通过调用 Android Support Library 中的底部导航栏组件 (BottomNavigationView) 类实现。将一些复杂逻辑丢给 Presenter 及相应的 Fragment，Activity 类则通过重载一些方法及实现一些接口，主要负责 Fragment 的初始调用和界面布局的可视化。

下面对该模块中一些重要的方法简要描述其功能：

- void MainActivity::onCreate(Bundle savedInstanceState): 初始化 Presenter 及各个 Fragment，确定当前 Fragment 界面，获取签名 (用于新闻分享)；
- void MainActivity::onSaveInstanceState(Bundle outState): 保存当前 Activity 状态，主要用于各个 Fragment 的保存；
- void switchToHome(), void switchToFavorite(), switchToSettings(): interface，通过在 MainActivity 类中调用 getSupportFragmentManager() 实现；
- boolean switchNavigation(int id): interface，根据 id 完成各个 Fragment 之间的切换，在 MainPresenter 类中实现，使用 switchToHome，switchToFavorite，switchToSettings

三个接口。

3.4 分类列表

本模块负责分类 (channel) 列表的添加与删除。通过在主界面的 NewsFragment 中点击分类列表右侧的十字图标触发该 Activity。主体为两个 recycleview：channel_mine 和 channel_more。前者保存用户选中的话题，后者保存用户删去的话题，分别设置两个监听者 channelMineListener 和 channelMoreListener，当其中一者中的某个新闻分类被用户点击时更新当前 Activity 的 View。

ChannelPresenter 类中主要由管理列表显示界面的 mView，和负责将数据更新到分类列表的 mNewsManager 组成。后者通过调用 NewsManager 类的 addCategory/deleteCategory 方法完成。

ChannelActivity 和 ChannelPresenter 遵循 ChannelContract 给出的接口

```
public interface ChannelContract {
    interface View extends BaseView<Presenter> {
        void setMineChannel(List<String> channelList);
        void setMoreChannel(List<String> channelList);
        void addChannel(String channel);
        void removeChannel(String channel);
    }

    interface Presenter extends BasePresenter {
        void addChannel(String channel);
        void removeChannel(String channel);
    }
}
```

下面对该模块中一些重要的方法简要描述其功能：

- void ChannelContract.View::setMineChannel(List<String> channelList),
  void ChannelContract.View::setMoreChannel(List<String> channelList)
  根据用户当前选中的分类列表初始化 channel_mine 和 channel_more；
- void ChannelContract.View::addChannel(String channel),
  void ChannelContract.View::removeChannel(String channel)

根据用户在 ChannelActivity 中选择的话题添加/删除某个新闻分类，显示在当前界面中。

3.5 新闻

该模块负责显示新闻列表 (newslist) 和新闻详情 (newsdetail) 的具体设计。其中，newsfragment 类表示 newslist 所在的 fragment 类，包括了搜索栏，分类列表与新闻列表。newslistadapter 类用于调节列表中每条新闻的显示方式（夜间模式，图片显示与否）。

3.5.1 新闻列表

新闻列表 (newslist) 的功能是显示相应的 Presenter 给出的新闻列表，通过 NewsListFragment 和 NewsListPresenter 类实现，易于复用。在应用中显示不同种类的新闻、收藏的新闻以及新闻搜索结果列表都是通过该类或其子类实现的。

NewsListFragment 和 NewsListPresenter 遵循 NewsListContract 给出的接口

```
public interface NewsListContract {
    interface View extends BaseView<Presenter> {
        void onSuccess(List<INewsIntroduction> newsList);
        void onFailure();
    }

    interface Presenter extends BasePresenter {
        void loadNewsList(int size, int pageNo, int category);
        void setFilter(INewsFilter filter);
    }
}
```

下面对该模块中一些重要的方法简要描述其功能：

- void NewsListContract.View::onSuccess(List<INewsIntroduction> newsList),
  void NewsListContract.View::onFailure()
  根据是否成功获取更多新闻加载新闻列表或显示错误信息；
- void NewsListContract.Presenter::loadNewsList(int size, int pageNo, int category)
  按照传入参数异步加载新闻列表；

- void NewsListContract.Presenter::setFilter(INewsFilter filter)
  设置 filter 用于新闻屏蔽功能。

3.5.2 新闻详情

新闻详情 (newsdetail) 的功能是显示用户选中新闻的详细信息，通过 NewsDetailActivity 和 NewsDetailPresenter 类实现。

NewsDetailActivity 和 NewsDetailPresenter 遵循 NewsDetailContract 给出的接口：

```
public interface NewsDetailContract {
    interface View extends BaseView<Presenter> {
        void onSuccess(INewsDetail newsDetail);
        void onFailure();
        void setLike();
        void setUnLike();
        void share(INewsDetail newsDetail);
    }

    interface Presenter extends BasePresenter {
        void loadNewsDetail(String newsId, boolean isTextOnly);
        void onLikeButtonClick();
        void onReadButtonClick();
        void onShareButtonCLick();
        void stopReading();
    }
}
```

下面对该模块中一些重要的方法简要描述其功能：

- void NewsDetailContract.View::onSuccess(INewsDetail newsDetail),
  void NewsDetailContract.View::onFailure() 根据是否成功获取新闻详情加载新闻详情或显示错误信息；
- void NewsDetailContract.View::setLike(),
  void NewsDetailContract.View::setUnLike()
  将当前新闻加入收藏或移除收藏；

- void NewsDetailContract.Presenter::loadNewsDetail(String newsId, boolean isTextOnly)
  根据传入的新闻 id 加载新闻；

- void NewsDetailContract.Presenter::onLikeButtonClick(),
  void NewsDetailContract.Presenter::onReadButtonClick(),
  void NewsDetailContract.Presenter::onShareButtonCLick()
  根据用户对按钮的点击调用 Model 执行相应操作；

3.6 收藏

收藏用于显示用户已收藏的新闻列表，通过 FavoriteNewsFragment 和 FavoriteNewsPresenter 类实现，FavoriteNewsFragment 继承了 NewsListFragment，FavoriteNewsPresenter 继承了 NewsListPresenter。除了将父类中向 Model 请求新闻列表改为向 Model 请求收藏列表，其余无变动。

该模块负责显示用户所有收藏的新闻列表。为了减少整体框架中 MainActivity 的负担和防止其过于臃肿，我们将 Favorite 设计为 Fragment。用户通过点击主界面下方的 Favorite 图案，使得 MainActivity 中的 Presenter 监听到切换信号，通过调用 MainContract.View::switchToSettings() 切换到 Favorite 界面，保持 UI 的通畅性。

FavoriteNewsPresenter 类继承自 NewsList 子包中的 NewsListPresenter 类，mNewsList 通过底层接口 NewsManager.getInstance().getFavoriteNews() 获取所有收藏新闻。FavoriteNewsFragment 类用来创建 FavoriteNewsPresenter 类的实例，最后通过 FavoriteFragment 类可视化包含用户收藏新闻的整个界面。FavoriteNewsFragment 作为 FavoriteFragment 的一个子 Fragment（只包含新闻），通过 Fragment 类中的 getChildFragmentManager 方法管理。

3.7 搜索

该模块负责显示用户在主界面搜索栏输入关键词后的新闻搜索结果。我们将新闻搜索结果设计为一个新的 Activity：SearchActivity 类，在用户在搜索栏输完关键词后，NewsFragment 类调用 SearchActivity 类的 start 方法，根据关键词的内容初始化一个符合用户搜索要求的结果界面。

类似于 Favorite 子包，SearchResultPresenter 类继承自 NewsList 子包中的 NewsListPresenter 类，mNewsList 通过底层接口 NewsManager.getInstance().searchNews(String query, int mode) 来获取所有满足搜索条件的新闻。SearchResultFragment 类用来创建 SearchResultPresenter 类的实例，然后通过 SearchActivity 类可视化包含搜索结果新闻的整个界

面。

3.8 设置

设置使用了 Android Support Library 提供的 PreferenceFragmentCompat，从 XML 文件中自动生成一个设置界面，不再赘述。

4 总结与心得