

知识点：

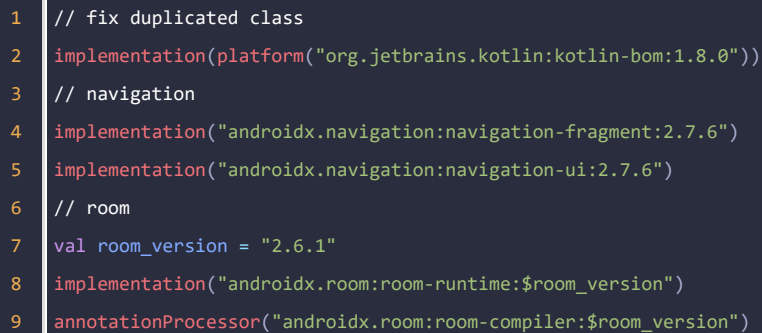
Fragment + Navigation + Room + SharedPreferences + ViewModel + LiveData + RecyclerView +  
OptionsMenu + PopupMenu + Intent + 消息框(Toast) + 对话框(AlertDialog) + 控件(View) + 界面布局  
(ViewGroup) + 事件监听

具体地，

1. 创建三个Fragments，并通过Navigation实现正确的页面导航（包括点击按钮和点击back键）
2. 通过Room实现数据库管理
3. 通过SharedPreferences实现数据存储
4. 通过ViewModel管理UI数据
5. 通过LiveData/MutableLiveData实现数据观测
6. 通过RecyclerView创建滚动列表
7. 通过Intent启动另一个Activity
8. 通过OptionsMenu设置RecyclerView整体的显示模式
9. 通过PopupMenu修改或删除RecyclerView中的单个Item
10. 通过创建消息框Toast弹出提示信息
11. 通过创建对话框AlertDialog进行二次确认
12. 通过控件+布局设计界面
13. 对用户的点击、长按、输入内容变化、焦点变化等事件添加监听，实现和用户的交互

步骤：

1. 添加Navigation和Room的依赖



```
1 // fix duplicated class
2 implementation(platform("org.jetbrains.kotlin:kotlin-bom:1.8.0"))
3 // navigation
4 implementation("androidx.navigation:navigation-fragment:2.7.6")
5 implementation("androidx.navigation:navigation-ui:2.7.6")
6 // room
7 val room_version = "2.6.1"
8 implementation("androidx.room:room-runtime:$room_version")
9 annotationProcessor("androidx.room:room-compiler:$room_version")
```

2. 创建三个Fragment：HomeFragment、WordsFragment、AddFragment

3. 设计三个Fragment的界面布局

- HomeFragment对应的布局文件

- 首先将FrameLayout改成ConstraintLayout，然后添加TextView、ImageView和Button，并修改控件id

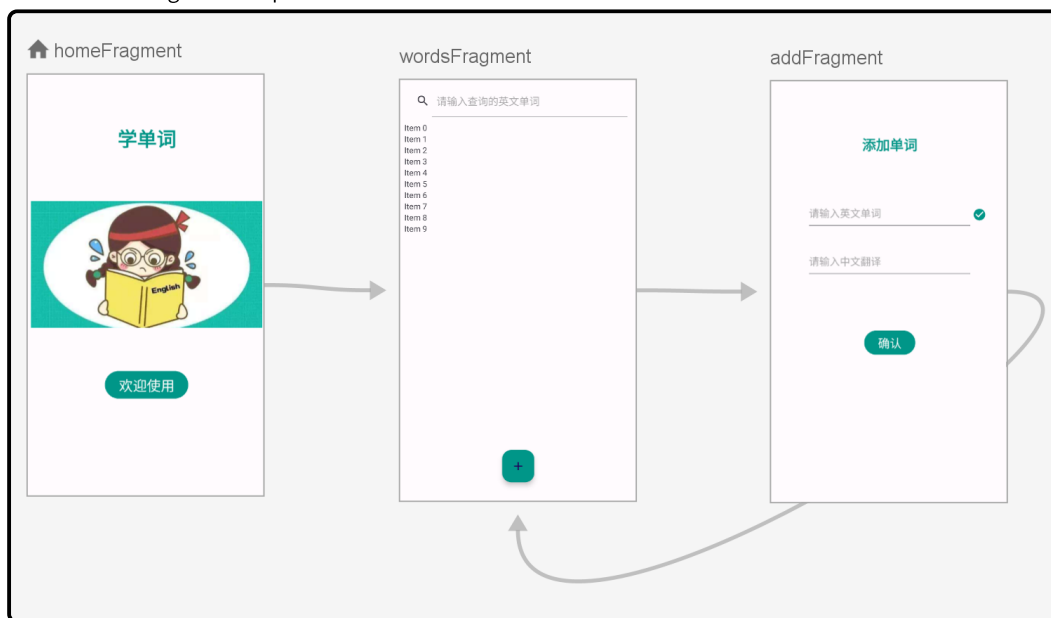
- WordsFragment对应的布局文件

- 首先将FrameLayout改成ConstraintLayout，然后添加SearchView、RecyclerView和FloatingActionButton，并修改控件id

- AddFragment对应的布局文件

- 首先将FrameLayout改成ConstraintLayout，然后添加TextView、EditText、Button和ImageView，并修改控件id

#### 4. 创建导航图 Navigation Graph

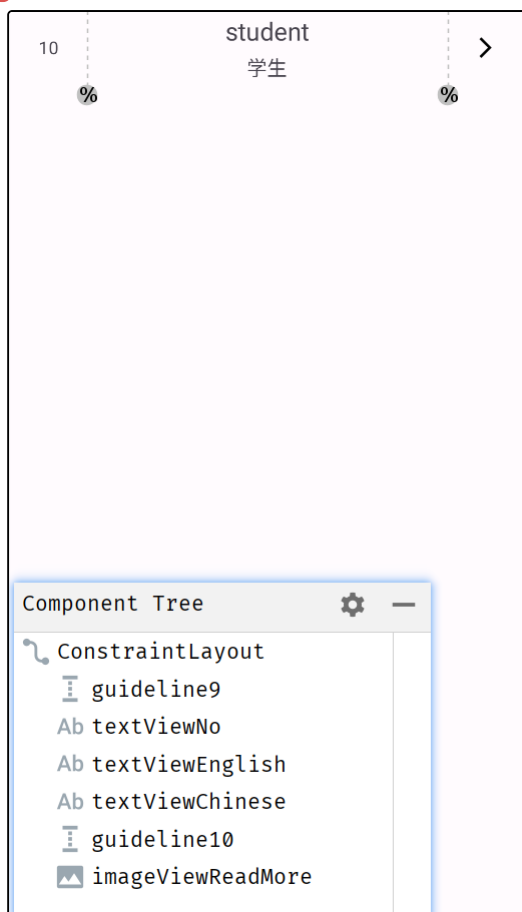


5. 在activity\_main.xml中添加NavHostFragment，并选择使用的导航图
6. 在各个Fragment中对各按钮实现事件监听，并在事件监听器中获取NavController实现页面导航

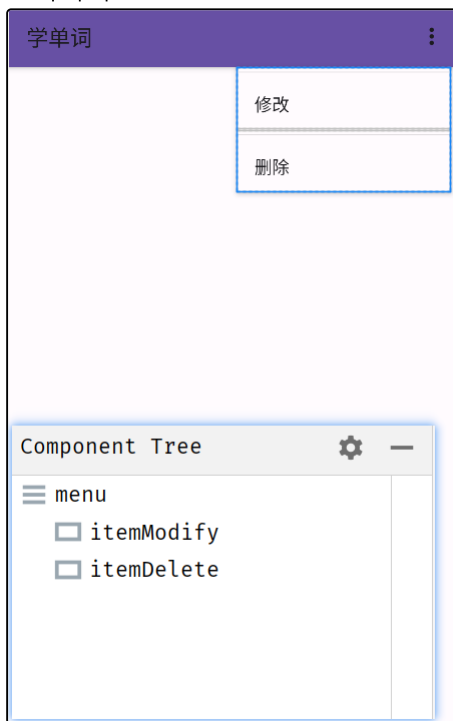
- 事件监听写在onCreateView()内
- 此处，可以运行检查导航是否正确

#### 7. 创建item\_view.xml并设计其界面布局

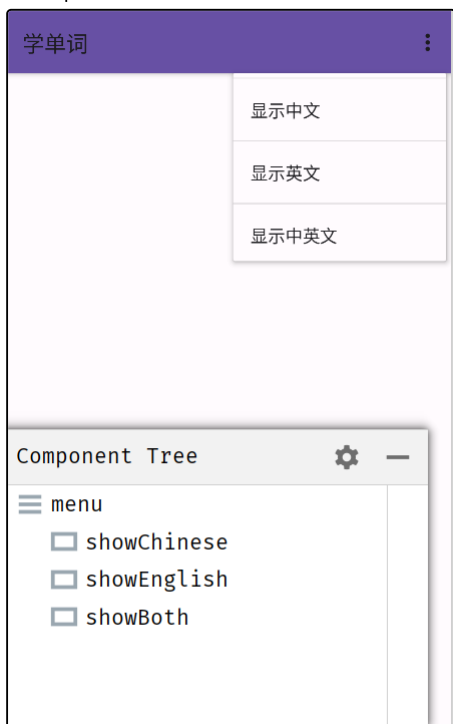
- 设置item “可点击、可长击” 以及 “点击效果”



#### 8. 创建popupMenu的布局文件



#### 9. 创建optionsMenu的布局文件



结合recyclerView和optionsMenu可以看出，需要存储两类数据：

- 一类是单词，用SQLite数据库存储
- 另一类是显示模式（即显示中文/英文/中英文），用SharedPreferences存储

#### 10. 修改Theme，采用有ActionBar的主题，并修改ActionBar上左上角的文字

#### 11. 创建Word、WordDao、WordDatabase

- Word包括：私有属性、构造方法、getter和setter方法，记得@Entity、@PrimaryKey等注解
- WordDao是接口，包括了对数据库的增删改查各操作的声明，记得@Dao、@Query等注解
- WordDatabase是抽象类，包括了一个构造方法和一个获取WordDao的抽象方法，采用单例模式，记得@Database注解

#### 12. 创建自定义的AndroidViewModel，用于管理UI数据，并对WordDao的方法进行封装

- UI数据包括“单词列表”和“显示模式”
  - “单词列表”的数据类型为LiveData<List < Word>>, 可实现数据观测, 用SQLite数据库存储, 所以AndroidViewModel需要包括相关方法
  - “显示模式”的数据类型为MutableLiveData < String>, 可实现数据观测, 用SharedPreferences存储, 所以AndroidViewModel需要包括相关方法

### 13. 创建自定义的RecyclerViewAdapter和ViewHolder。

- RecyclerViewAdapter的属性包括: AndroidViewModel实例、单词列表(List<Word>)和显示模式(String)。
- RecyclerViewAdapter需要重写onCreateViewHolder、onBindViewHolder和getItemCount方法。
- 在onBindViewHolder方法中进行数据绑定和事件监听, 其中:
  - 根据显示模式设置控件是否可见
  - 监听图片的点击事件, 并在监听器中实现通过Intent打开指定网页
  - 监听itemView的长按事件, 并在监听器中创建popupMenu, 并实现popupMenu中的修改、删除功能
    - 修改单词时获取NavController实现页面跳转, 并传入参数(区别于添加单词)
    - 删除前需要弹出对话框确认

### 14. 修改WordsFragment

- 获取AndroidViewModel实例
- 设置RecyclerView的Adapter和LayoutManager
- 实现对LiveData的数据观测, 包括“单词列表”(LiveData<List < Word>>)和“显示模式”(MutableLiveData < String>)
- 适时读取和保存SharedPreferences中的“显示模式”数据
- 添加FloatingActionButton的事件监听, 在监听器中获取NavController实现页面跳转, 并传入参数(区别于修改单词)
- 添加对SearchView的事件监听, 并适时更新“单词列表”及其观测
- 创建optionsMenu, 并实现菜单项的事件监听

### 15. 修改AddFragment

- 获取AndroidViewModel实例
- 根据传入的参数区别对待“添加单词”和“修改单词”
  - 添加单词时, 保证输入内容不为空且不能重复添加已存在的单词
  - 修改单词时, 主键(English)不能修改
- 添加/修改单词成功后, 弹出消息框, 并获取NavController实现页面跳转

### 16. 修改MainActivity

- 重写onSupportNavigateUp()和onBackPressed(), 重新设置点击back键的功能
  - 当在HomeFragment上点击back键时, 弹出对话框, 实现退出程序
  - 当在WordsFragment上点击back键时, 跳转到HomeFragment
  - 当在AddFragment上点击back键时, 跳转到WordsFragment

