智能移动开发课程报告

1813041　　杨晖

需求分析

1.1 项目介绍

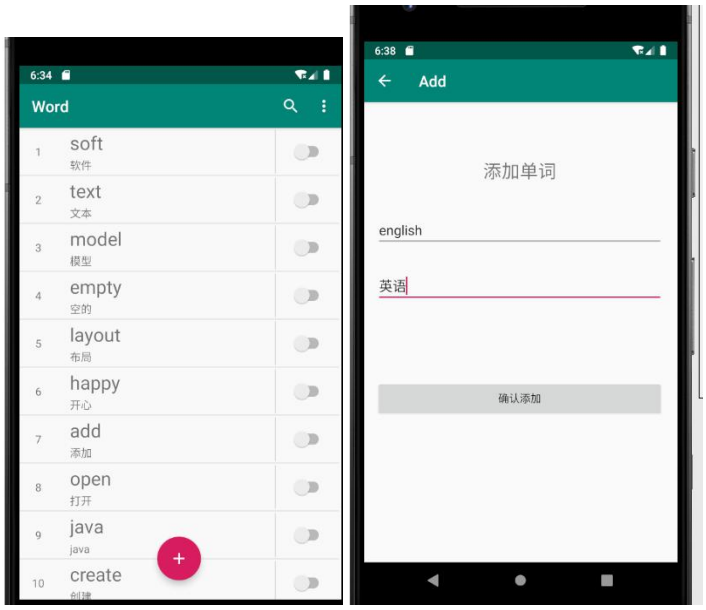
文化交流之间最大的障碍无疑就是帮助相互了解的语言了，有感于本人英语水平，开发了一个 app。此项目是一个帮助记忆单词的单词本，用户输入要记忆的单词，系统会自动保存到 sqllte 数据库中，用户可以自行设置界面风格。设置单词记忆顺序等，可以查找单词，删除单词，以及隐藏中文。可以帮助用户更好的记忆单词。

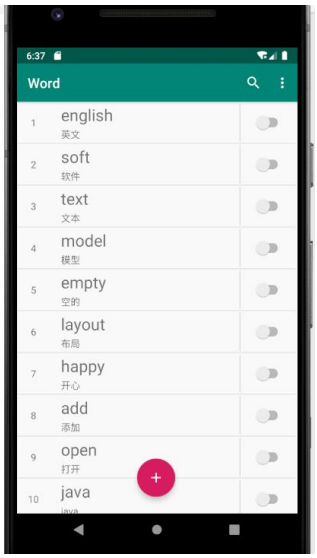
1.2 功能需求

1. 单词本可以通过用户输入中英文，来实现单词的添加。
2. 用户可以随意开关某个单词的中文释义
3. 用户可以删除已经添加的单词
4. 用户关机或者退出应用不会导致单词本内的单词丢失
5. 用户可以自定义单词本中单词的顺序
6. 可以选择自己喜欢的界面风格
7. 用户可以快捷的查找某个单词在有道上的详细信息
8. 用户可以查找某个单词

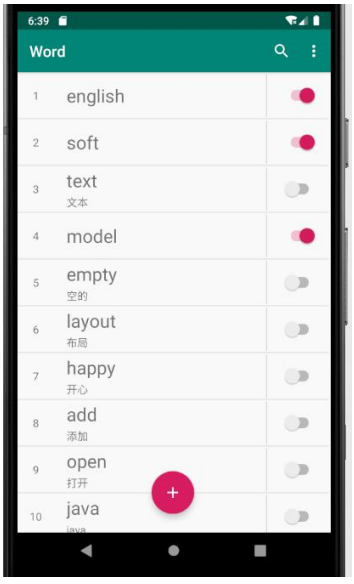
1.3 用法及规则

1. 用户点击加号按钮输入单词，键盘会自动弹出并焦点在输入框，用户输入英文和中文释义，单词本上将在最上面显示这个单词，并会有下拉动画。用户可以通过上方放回箭头取消输入。

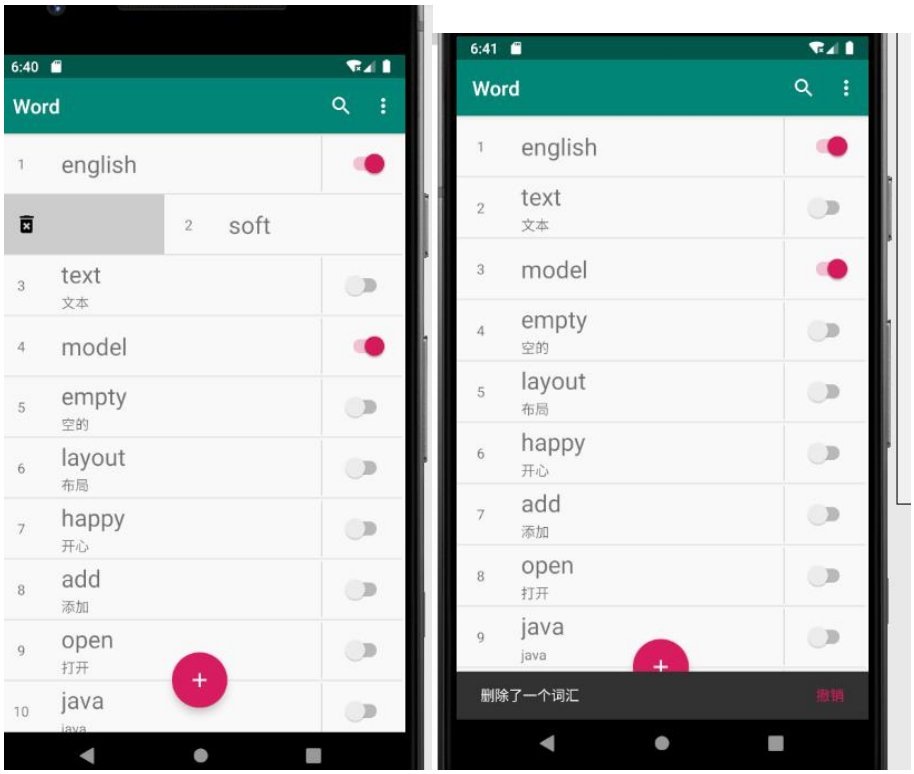




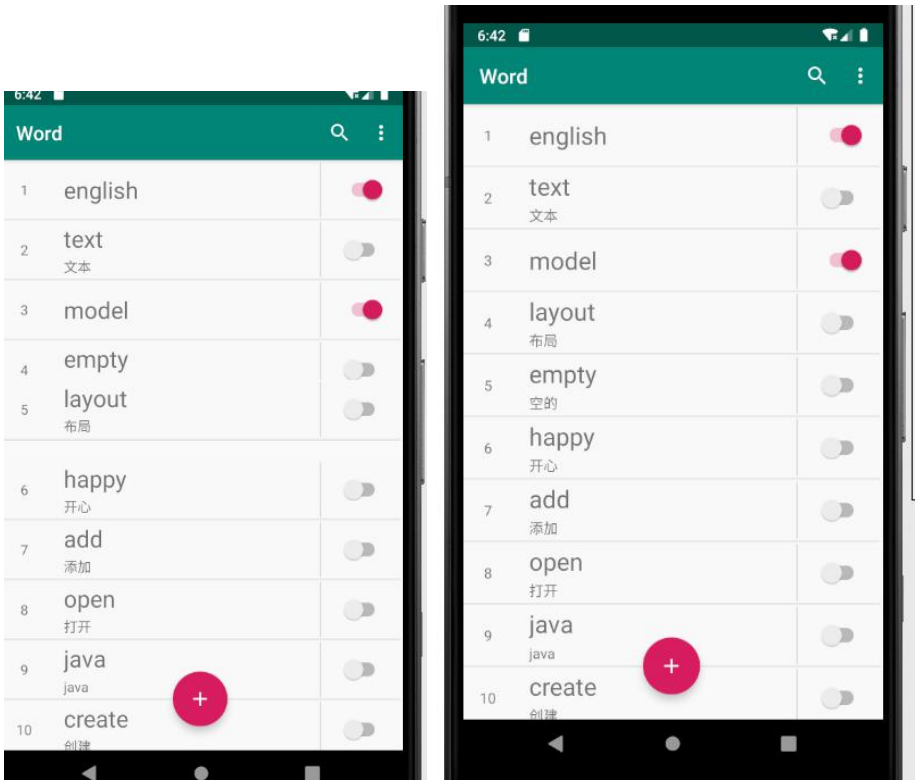
2. 用户可以通过单词盘边的按钮来关闭或开启中文意思



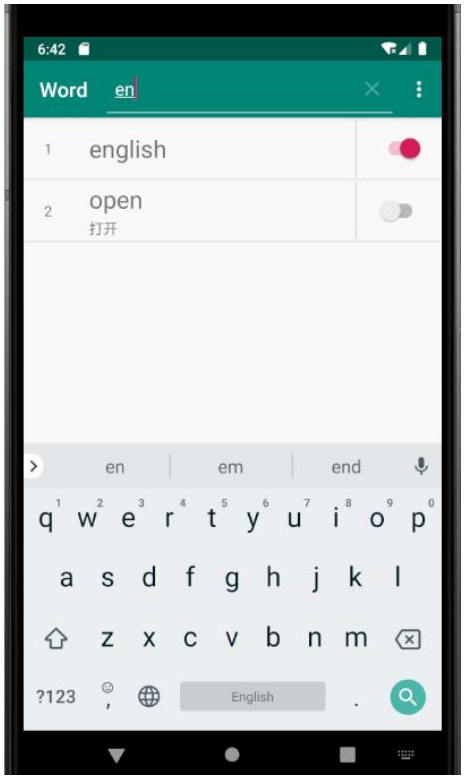
3. 向右滑可以删除单词，同时底部有提示信息，可以撤销删除，点击撤销可以还原被删除的单词



4. 长按拖住上下滑动可以设置单词的优先级

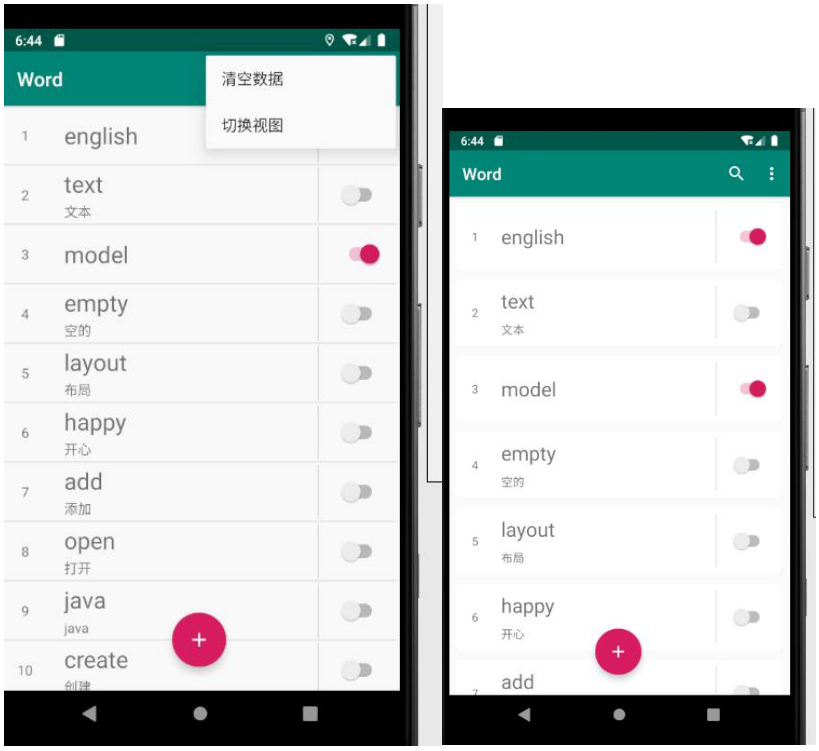


5. 搜索可以搜索相关的单词



6. 菜单可以选择切换界面或者删除全部单词，将以卡片/列表显示单词

7. 单击已经录入的单词会自动在有道上搜索



一、 系统设计

2.1 本程序需解决的有关技术问题

1. CardView 和 constraintLayout 的布局切换

2.通过继承 asyncTask 实现多线程查询数据库

3.查询数据库过程中多个类的调用和封装解耦，模块化开发

4.通过 livedate 监听数据变化来改变视图显示。

5.如何通过滑动删除单词和如何改变单词位置

6.各个 framgement 之间的切换导航

2.2 程序流程

1. 程序启动后先通过 MainActivity 的 onCreate 方法

2. 进入 wordsFramgment 类，通过 MyAdaper new 出单词列表

3. 当通过监听 WordViewModel 发现数据发生改变时，会同时增删改单词列表条数

4. 对数据改变的方法通过 WordViewModel 类中的方法调用 WordReposity 中的方法再调用 WordDatabase 或 WordDao 中的方法进行增删改或者查询

5.通过 Word 建立数据库表和映射关系。

二、 程序实现

3.1 类分析与设计

1）Word 类：

1.这个这个类定义了单词条目的属性：id word chineseWord chineseInvisible 并同时映射到数据库对应的属性字段；

2.定义了 set/get 方法用于存取数据

```
@Entity
public class Word {
    @PrimaryKey(autoGenerate = true)
    private int id;
    @ColumnInfo(name = "english_word")
    private String word;
    @ColumnInfo(name = "chinese_meaning")
    private String chineseMeaning;
    @ColumnInfo(name = "chinese_invisible")
    private boolean chineseInvisible;

    public Word(String word, String chineseMeaning
        this.word = word;
        this.chineseMeaning = chineseMeaning;
    }

    public boolean isChineseInvisible() { return c

    public void setChineseInvisible(boolean chines
        this.chineseInvisible = chineseInvisible;
```

2）WordDao 类：

1.dao 层用注解定义了增删改查方法

对于复杂查询，再注解中定义了 sql 语句：

A.
```
@Query("select * from word order by id desc")
```

B.模糊查询并按倒叙排序

```
@Query("select * from word where english_word like :patten order by id desc ")
```

```
@Dao
public interface WordDao {
    @Insert
    void insertWords(Word... words);
    @Update
    void updateWords(Word... words);
    @Delete
    void deleteWords(Word... words);

    @Query("delete from word")
    void deleteAllWords();

    @Query("select * from word order by id desc")
    LiveData<List<Word>> getAllWordsLive();

    @Query("select * from word where english_word like :patten order by id desc ")
    LiveData<List<Word>> findWordsWithPattern(String patten);
```

3）WordDatabase 类：

1.利用单例工厂模式，new 了一个 WordDataBase 类，这个类继承了 RoomDatabase，建立了数据库，用来通过 getDatabase 来实例化 WordDao

```
public abstract class WordDatabase extends RoomDatabase {
    private static WordDatabase INSTANCE;
    static synchronized WordDatabase getDatabase(Context context){
        if(INSTANCE == null ){
            INSTANCE = Room.databaseBuilder(context.getApplicationContext(),WordDatabase.class, name: "words_database")
                    .addMigrations(MIGRATION_4_5)
                    .build();
        }
        return INSTANCE;
    }
```

Static 可以不需要 new 就可以调用方法

Synchronized 同步块只有一个再执行，反正多线程而 new 了多个实例

此方法构建并返回一个 wordDao。同此此方法中还包含数据库的升级及修改方法

4）WordRepository 类：

1.多线程调用 WordDao 的方法完成增删改查。

```
public LiveData<List<Word>> getAllWordsLive() { return allWordsLive; }
public LiveData<List<Word>> findWordsWithPattern(String patten){
    return wordDao.findWordsWithPattern( patten: "%" + patten + "%");
}

void insertWords(Word... words) { new InsertAsyncTask(wordDao).execute(words); }
void updateWords(Word... words) { new UpdateAsyncTask(wordDao).execute(words); }
void deleteWords(Word... words) { new DeleteAsyncTask(wordDao).execute(words); }
void deleteAllWords(Word... words) { new DeleteAllAsyncTask(wordDao).execute(); }
```

2. 主要功能是实现多线程

```
static class InsertAsyncTask extends AsyncTask<Word, Void, Void> {
    private WordDao wordDao;

    public InsertAsyncTask(WordDao wordDao) { this.wordDao = wordDao; }

    @Override
    protected Void doInBackground(Word... words) {
        wordDao.insertWords(words);
        return null;
    }
}

static class UpdateAsyncTask extends AsyncTask<Word, Void, Void>{
    private WordDao wordDao;

    public UpdateAsyncTask(WordDao wordDao) { this.wordDao = wordDao; }

    @Override
    protected Void doInBackground(Word... words) {
        wordDao.updateWords(words);
        return null;
    }
}

static class DeleteAsyncTask extends AsyncTask<Word, Void, Void>{
    private WordDao wordDao;
```

5）WordViewModel 类：

1.通过调用 WordRepository 类的方法完成增删改查

```
void insertWords(Word... words) { wordRepository.insertWords(words); }
void updateWords(Word... words) { wordRepository.updateWords(words); }
void deleteWords(Word... words) { wordRepository.deleteWords(words); }
void deleteAllWords() { wordRepository.deleteAllWords(); }
```

2.主要功能是处理数据相关功能，返回的 LiveData 可以监控数据变化，并再变化时调用相应方法来处理视图等。

```
public LiveData<List<Word>> getAllWordsLive() { return wordRepository.getAllWordsLive(); }
public LiveData<List<Word>> findWordsWithPattern(String patten){
    return wordRepository.findWordsWithPattern(patten);
}
```

7）MainActivity 类：

1.设置了视图导航。

```
navController = Navigation.findNavController((findViewById(R.id.fragment)));
NavigationUI.setupActionBarWithNavController( activity: this,navController);
```

2.重写了返回键的方法

```
@Override
public boolean onSupportNavigateUp() {
    InputMethodManager imm = (InputMethodManager) this.getSystemService(Context.INPUT_METHOD_SERVICE);
    imm.hideSoftInputFromWindow(findViewById(R.id.fragment).getWindowToken(), flags: 0);
    navController.navigateUp();
    return super.onSupportNavigateUp();
}
```

8）AddFragment 类：

1. 定义了添加单词界面的功能
2. 进入界面后，将输入框聚焦在英语输入框上，并弹出键盘。

```
editTextEnglish.requestFocus();
InputMethodManager imm = (InputMethodManager)activity.getSystemService(Context.INPUT_METHOD_SERVICE);
imm.showSoftInput(editTextEnglish, flags: 0);
```

3. 点击添加按钮后，通过 WordViewModel 将数据写入数据库，并通过导航回到主界面

```
buttonSubmit.setOnClickListener((v) -> {
        String english = editTextEnglish.getText().toString().trim();
        String chinese = editTextChinese.getText().toString().trim();
        Word word = new Word(english,chinese);
        wordViewModel.insertWords(word);
        NavController navController = Navigation.findNavController(v);
        navController.navigateUp();
```

9）MyAdapter 类

1. 通过点击单词条，用 intent 并传入一个网址可以跳到有道词典搜索

```
holder.itemView.setOnClickListener((v) -> {
        Uri uri = Uri.parse("https:///m.youdao.com/dict?le=eng&q=" + holder.textViewEnglish.getText());
        Intent intent = new Intent(Intent.ACTION_VIEW);
        intent.setData(uri);
        holder.itemView.getContext().startActivity(intent);
});
```

2. 通过查询数据库 ChineseInvisible 字段，初始化设置是否显示中文释义

```
if(word.isChineseInvisible()){
    holder.textViewChinese.setVisibility(View.GONE);
    holder.aSwitchChineseInvisible.setChecked(true);
}else {
    holder.textViewChinese.setVisibility(View.VISIBLE);
    holder.aSwitchChineseInvisible.setChecked(false);
}
```

10）WordFragment 类

1. 搜索功能

```
ublic void onCreateOptionsMenu(@NonNull Menu menu, @NonNull MenuInflater inflater) {
    super.onCreateOptionsMenu(menu, inflater);
    inflater.inflate(R.menu.main_enu, menu);
    SearchView searchView = (SearchView) menu.findItem(R.id.app_bar_search).getActionView();
    searchView.setMaxWidth(780);
    searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
        @Override
        public boolean onQueryTextSubmit(String query) { return false; }

        @Override
        public boolean onQueryTextChange(String newText) {
            String patten = newText.trim();
            filteredWords.removeObservers(requireActivity());
            filteredWords = wordViewModel.findWordsWithPattern(patten);
            filteredWords.observe(getViewLifecycleOwner(), new Observer<List<Word>>() {
                @Override
                public void onChanged(List<Word> words) {
                    int temp = myAdapter1.getItemCount();
                    allWords = words;
                    if (temp != words.size()) {
                        myAdapter1.submitList(words);
                        myAdapter2.submitList(words);
                    }
                }
            });
            return true;
```

3.在 WordsFragment 中设置了菜单功能，点击清空数据会弹出 alertDialog 对话框，若选择确认，则调用 wordViewModel 中的 deleteAll（）方法删除所以数据；点击切换视图时，会先在 SharePreferences 中获得现在时什么视图的值，然后调用 recyclerView 的方法设置另一个视图。

```
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId()) {
        case R.id.clearData:
            AlertDialog.Builder builder = new AlertDialog.Builder(requireActivity());
            builder.setTitle("清空数据");
            builder.setPositiveButton( text: "确定", (dialog, which) -> {
                    wordViewModel.deleteAllWords();
            });
            builder.setNegativeButton( text: "取消", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {

                }
            });
            builder.create();
            builder.show();
            break;
        case R.id.switchViewType:
            SharedPreferences shp = requireActivity().getSharedPreferences(VIEW_TYPE_SHP, Context.MODE_PRIVATE);
            boolean viewType = shp.getBoolean(IS_USING_CARD_VIEW, defValue: false);
            SharedPreferences.Editor editor = shp.edit();
            if (viewType) {
                recyclerView.setAdapter(myAdapter1);
                recyclerView.addItemDecoration(dividerItemDecoration);
                editor.putBoolean(IS_USING_CARD_VIEW, false);
            } else {
                recyclerView.setAdapter(myAdapter2);
                recyclerView.removeItemDecoration(dividerItemDecoration);
                editor.putBoolean(IS_USING_CARD_VIEW, true);
            }
            editor.apply();
    }
    return super.onOptionsItemSelected(item);
```

3. 用 onChildDraw 画滑动后面的灰色阴影和垃圾桶图标

```
@Override
public void onChildDraw(@NonNull Canvas c, @NonNull RecyclerView recyclerView, @NonNull RecyclerView.ViewHolder viewHolder, float dX, float dY, int actionState,
    super.onChildDraw(c, recyclerView, viewHolder, dX, dY, actionState, isCurrentlyActive);
    View itemView = viewHolder.itemView;
    int iconMargin = ((itemView.getHeight()) - icon.getIntrinsicHeight())/2;
    int iconLeft, iconRight, iconTop, iconBottom;
    int backTop, backBottom, backLeft, backRight;
    backTop = itemView.getTop();
    backBottom = itemView.getBottom();
    iconTop = itemView.getTop() + (itemView.getHeight() - icon.getIntrinsicHeight()) /2;
    iconBottom = iconTop + icon.getIntrinsicHeight();
    if(dX > 0){
        backLeft = itemView.getLeft();
        backRight = itemView.getLeft() + (int)dX;
        background.setBounds(backLeft,backTop,backRight,backBottom);
        iconLeft = itemView.getLeft() + iconMargin;
        iconRight = iconLeft + icon.getIntrinsicWidth();
        icon.setBounds(iconLeft,iconTop,iconRight,iconBottom);
    }else if (dX < 0){
        backRight = itemView.getRight();
        backLeft = itemView.getRight() + (int)dX;
        background.setBounds(backLeft,backTop,backRight,backBottom);
        iconRight = itemView.getRight() + iconMargin;
        iconLeft = iconRight + icon.getIntrinsicWidth();
        icon.setBounds(iconLeft,iconTop,iconRight,iconBottom);
    }else {
        background.setBounds( left: 0, top: 0, right: 0, bottom: 0);
        icon.setBounds( left: 0, top: 0, right: 0, bottom: 0);
    }
    background.draw(c);
    icon.draw(c);
```

4.点击悬浮按钮跳到添加单词界面

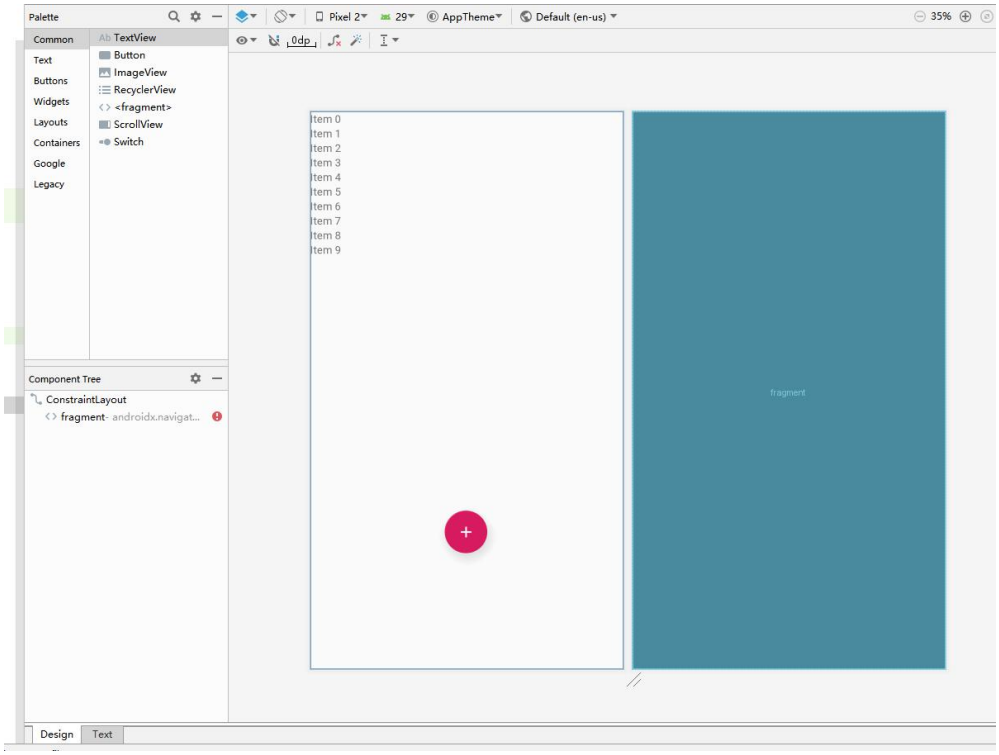
```
floatingActionButton = requireActivity().findViewById(R.id.floatingActionButton);
floatingActionButton.setOnClickListener((v) -> {
        NavController navController = Navigation.findNavController(v);
        navController.navigate(R.id.action_wordsFragment_to_addFragment);
});
```

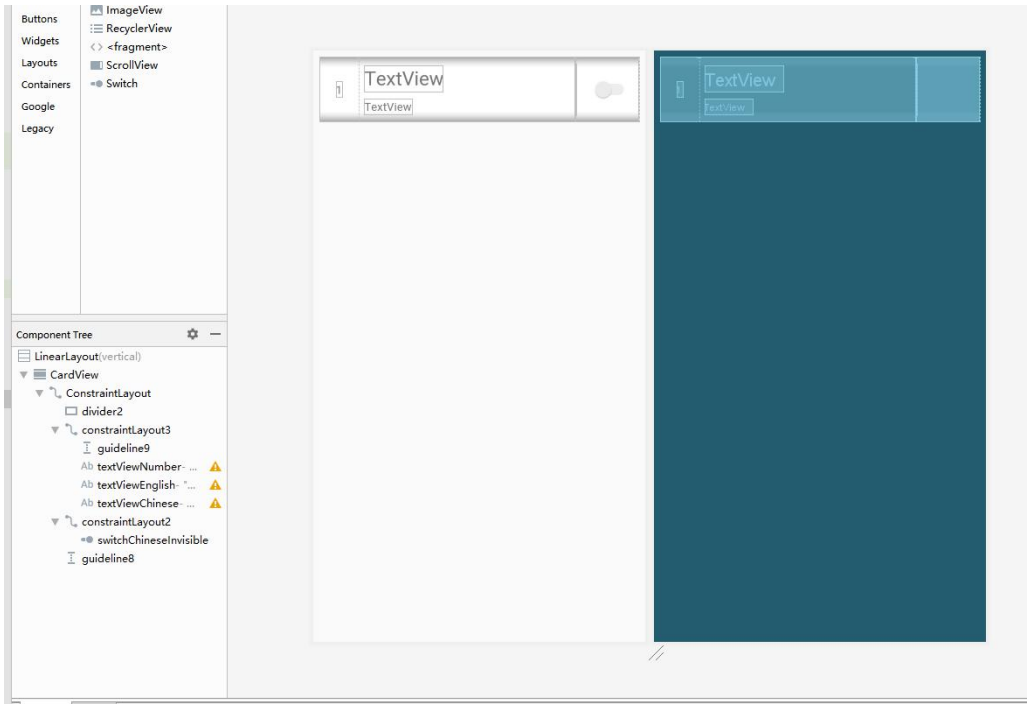
5.移动单词和滑动删除单词功能

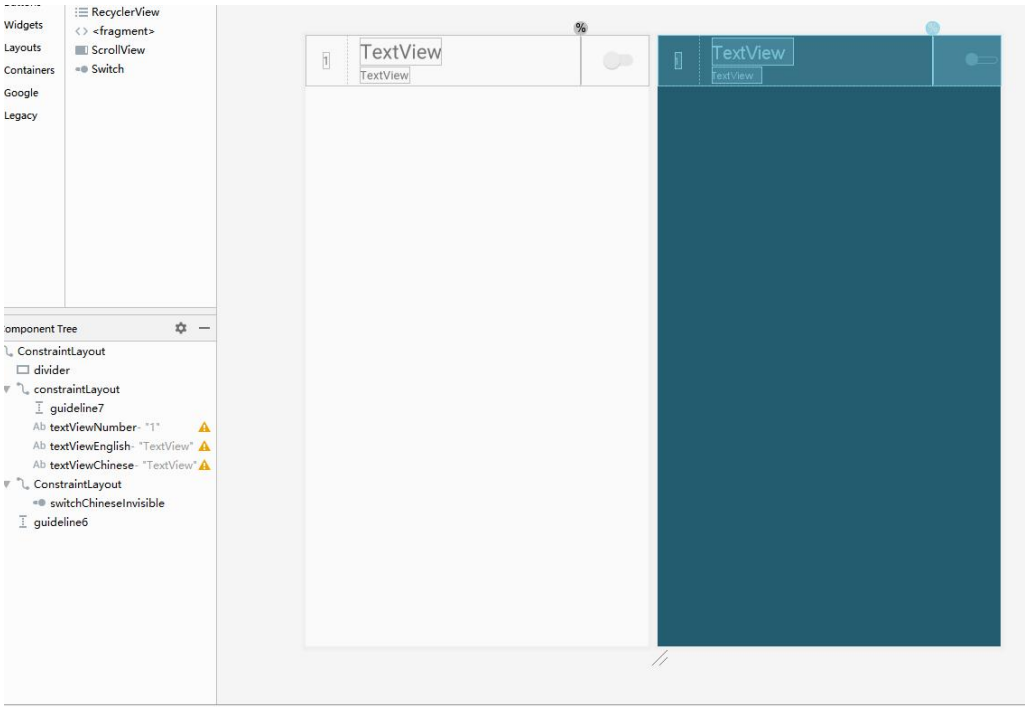
```
@Override
public boolean onMove(@NonNull RecyclerView recyclerView, @NonNull RecyclerView.ViewHolder viewHolder, @NonNull RecyclerView.ViewHolder target) {
    Word wordFrom = allWords.get(viewHolder.getAdapterPosition());
    Word wordTo = allWords.get(target.getAdapterPosition());
    int idTemp = wordFrom.getId();
    wordFrom.setId(wordTo.getId());
    wordFrom.setId(idTemp);
    wordViewModel.updateWords(wordFrom,wordTo);
    myAdapter1.notifyItemMoved(viewHolder.getAdapterPosition(),target.getAdapterPosition());
    myAdapter2.notifyItemMoved(viewHolder.getAdapterPosition(),target.getAdapterPosition());
    return false;
}

@Override
public void onSwiped(@NonNull RecyclerView.ViewHolder viewHolder, int direction) {
    final Word wordToDelete = allWords.get(viewHolder.getAdapterPosition());
    wordViewModel.deleteWords(wordToDelete);
    Snackbar.make(requireActivity().findViewById(R.id.wordFragementView), text: "删除了一个词汇", Snackbar.LENGTH_SHORT)
            .setAction( text: "撤销", (v) -> {
                    undoAction = true;
                    wordViewModel.insertWords(wordToDelete);
            })
            .show();
}
```

3.2 界面布局设置

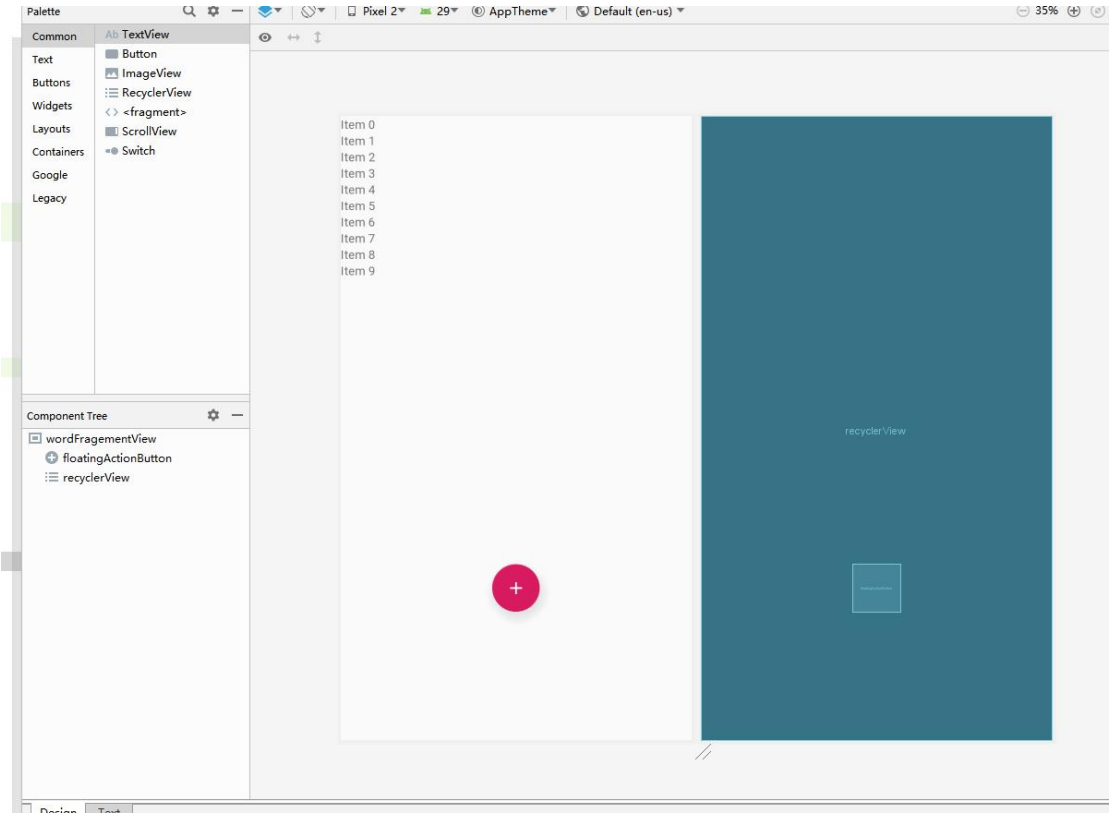
界面布局采用图形化的方式



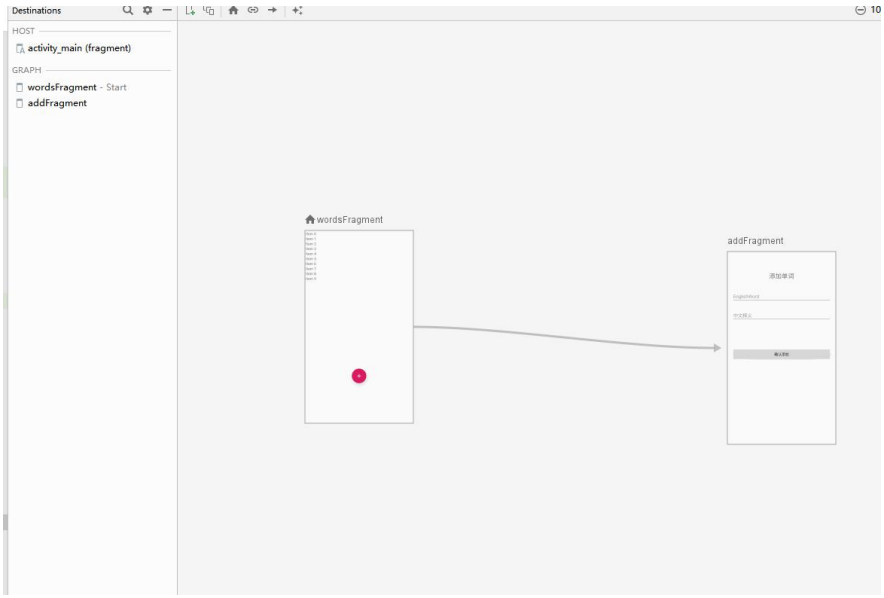



Widgets
Layouts
Containers
Google
Legacy
RecyclerView
<fragment>
ScrollView
Switch
omponent Tree
ConstraintLayout
divider
constraintLayout
guideline7
textViewNumber - "1"
textViewEnglish - "TextView"
textViewChinese - "TextView"
ConstraintLayout
switchChineseInvisible
guideline6
TextView
TextView
TextView
TextView

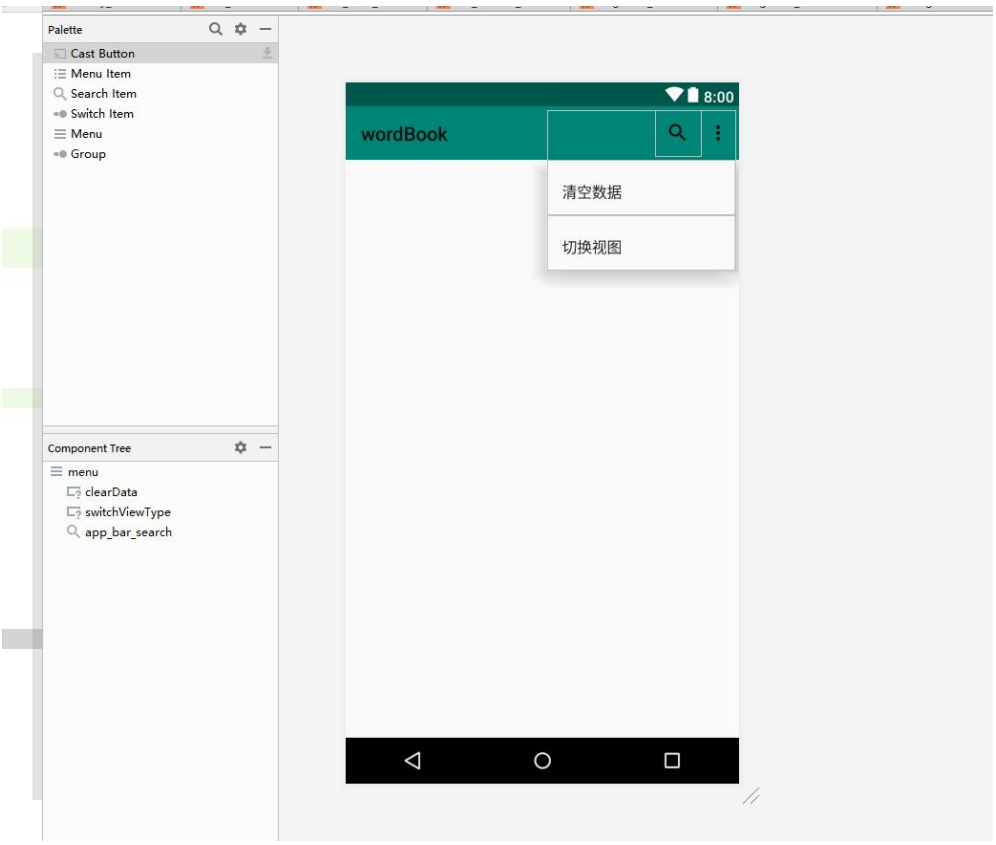

Buttons
Widgets
Layouts
Containers
Google
Legacy
ImageView
RecyclerView
<fragment>
ScrollView
Switch
Component Tree
addLayout
textView - "@string/addNewW...
guideline
guideline12
guideline11
guideline10
editTextEnglish (Plain Text)
editTextChinese (Plain Text)
buttonSubmit - "@string/Butto...
添加单词
EnglishWord
中文释义
确认添加
添加单词
editTextEnglish
editTextChinese
确认添加



3.3 导航设置



3.4 菜单设置



3.5 字符串资源文件

```
<resources>
    <string name="app_name">wordBook</string>

    <!-- TODO: Remove or change this placeholder text -->
    <string name="hello_blank_fragment">Hello blank fragment</string>
    <string name="addNewWord" >添加单词</string>
    <string name="EnglishWord">EnglishWord</string>
    <string name="chinese">中文释义</string>
    <string name="ButtonOk">确认添加</string>
</resources>
```

3.7 部分源码

1.在 AddFramgent 中，若输入框两个都不为空，设置添加按钮可用

```
@Override
public void onTextChanged(CharSequence s, int start, int before, int count) {
    String english = editTextEnglish.getText().toString().trim();
    String chinese = editTextChinese.getText().toString().trim();
    buttonSubmit.setEnabled(!english.isEmpty() && !chinese.isEmpty());
}
```

2.在 MyAdapter 中，监听开关，若发生改变显示/隐藏中文释义，同时修改数据库中 ChineseVisible 字段的值。

```
holder.aSwitchChineseInvisible.setOnCheckedChangeListener((buttonView, isChecked) -> {
        Word word = (Word)holder.itemView.getTag(R.id.word_for_view_holder);
        if(isChecked){
            holder.textViewChinese.setVisibility(View.GONE);
            word.setChineseInvisible(true);
            wordViewModel.updateWords(word);

        }else {
            holder.textViewChinese.setVisibility(View.VISIBLE);
            word.setChineseInvisible(false);
            wordViewModel.updateWords(word);
        }
});
```

3.在 WordsFragment 中设置了菜单功能，点击清空数据会弹出 alertDialog 对话框，若选择确认，则调用 wordViewModel 中的 deleteAll（）方法删除所以数据；点击切换视图时，会先在 SharePreferences 中获得现在时什么视图的值，然后调用 recyclerView 的方法设置另一个视图。

```java
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId()) {
        case R.id.clearData:
            AlertDialog.Builder builder = new AlertDialog.Builder(requireActivity());
            builder.setTitle("清空数据");
            builder.setPositiveButton( text: "确定", (dialog, which) -> {
                    wordViewModel.deleteAllWords();
            });
            builder.setNegativeButton( text: "取消", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {

                }
            });
            builder.create();
            builder.show();
            break;
        case R.id.switchViewType:
            SharedPreferences shp = requireActivity().getSharedPreferences(VIEW_TYPE_SHP, Context.MODE_PRIVATE);
            boolean viewType = shp.getBoolean(IS_USING_CARD_VIEW, defValue: false);
            SharedPreferences.Editor editor = shp.edit();
            if (viewType) {
                recyclerView.setAdapter(myAdapter1);
                recyclerView.addItemDecoration(dividerItemDecoration);
                editor.putBoolean(IS_USING_CARD_VIEW, false);
            } else {
                recyclerView.setAdapter(myAdapter2);
                recyclerView.removeItemDecoration(dividerItemDecoration);
                editor.putBoolean(IS_USING_CARD_VIEW, true);
            }
            editor.apply();
    }
    return super.onOptionsItemSelected(item);
```

ordsFragment › onActivityCreated()

4.滑动删除后会弹出一个 snackbar，点击即可撤销删除

```java
Snackbar.make(requireActivity().findViewById(R.id.wordFragementView), text: "删除了一个词汇", Snackbar.LENGTH_SHORT)
        .setAction( text: "撤销", (v) -> {
                undoAction = true;
                wordViewModel.insertWords(wordToDelete);
        })
        .show();
```

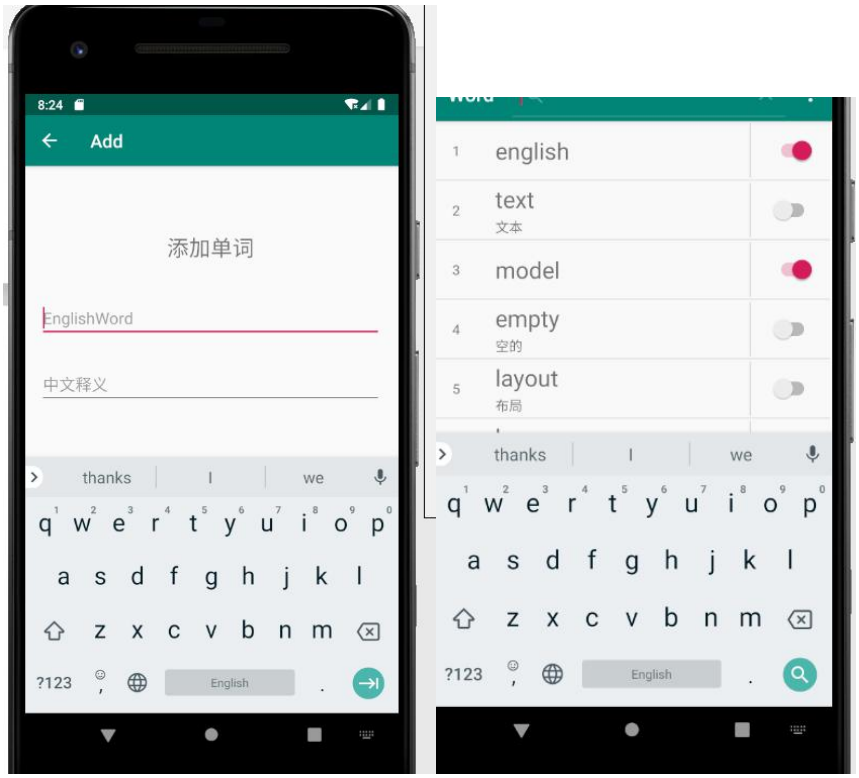
5.onMove（）方法处理长按拖动事件，发生拖动时，替换两个 Word 的 id，livedate 检测到数据发生改变，就重新绘画界面

```java
@Override
public boolean onMove(@NonNull RecyclerView recyclerView, @NonNull RecyclerView.ViewHolder viewHolder, @NonNull RecyclerView.ViewHolder target) {
    Word wordFrom = allWords.get(viewHolder.getAdapterPosition());
    Word wordTo = allWords.get(target.getAdapterPosition());
    int idTemp = wordFrom.getId();
    wordFrom.setId(wordTo.getId());
    wordFrom.setId(idTemp);
    wordViewModel.updateWords(wordFrom,wordTo);
    myAdapter1.notifyItemMoved(viewHolder.getAdapterPosition(),target.getAdapterPosition());
    myAdapter2.notifyItemMoved(viewHolder.getAdapterPosition(),target.getAdapterPosition());
    return false;
}
```

三、 系统测试

一、程序存在的问题

1. 再添加单词页面点击上分或下方的返回，键盘不会消失



解决方法：再 AddFagment 中 button 触发方法中添加键盘回缩方法：

```
navController.navigateUp();
InputMethodManager imm = (InputMethodManager)activity.getSystemService(Context.INPUT_METHOD_SERVICE);
imm.hideSoftInputFromWindow(v.getWindowToken(), flags: 0);
```

2. 再添加单词页面中，没有自动聚焦到英语单词输入框

解决方法：再 onActivityCreate 中添加方法

```
InputMethodManager imm = (InputMethodManager)activity.getSystemService(Context.INPUT_METHOD_SERVICE);
imm.showSoftInput(editTextEnglish, flags: 0);
```

3. 若长按拖动速度过快，会导致有些单词的中文意思会自动隐藏。

4. 列表界面单词之间的分界不够明显

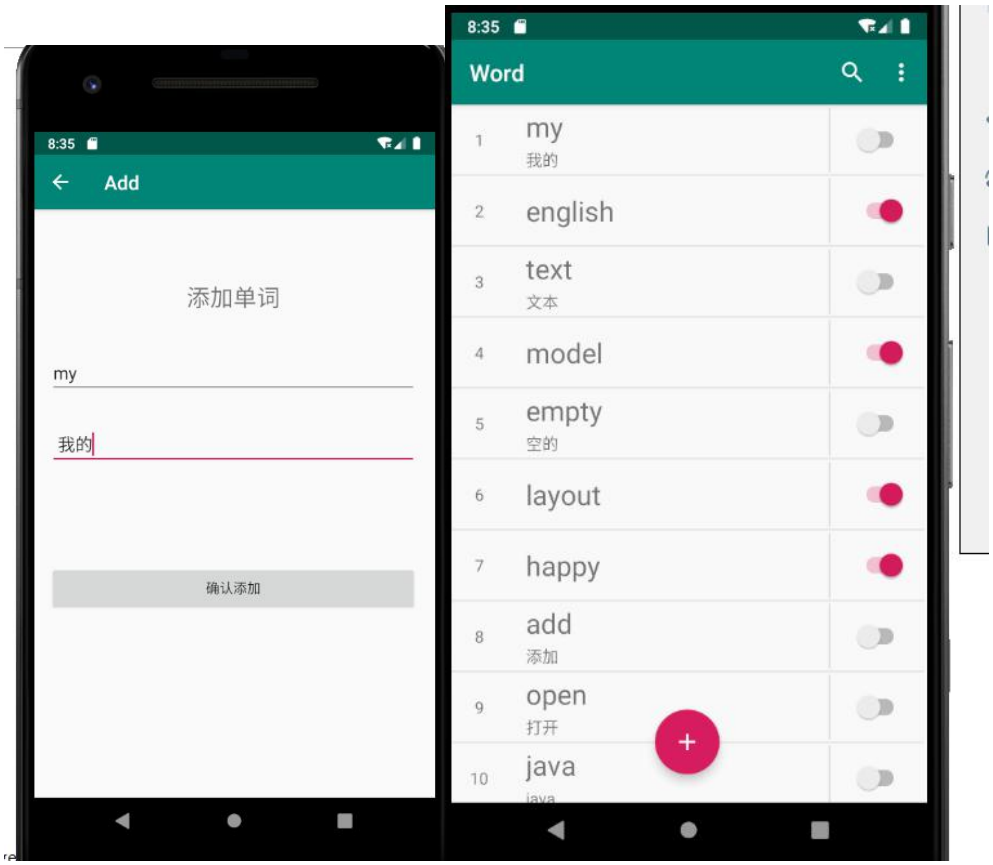
解决方法 再每个列表条目上添加下划线。

```
boolean viewType = shp.getBoolean(IS_USING_CARD_VIEW, defValue: false);
dividerItemDecoration = new DividerItemDecoration(requireActivity(),DividerItemDecoration.VERTICAL);
```

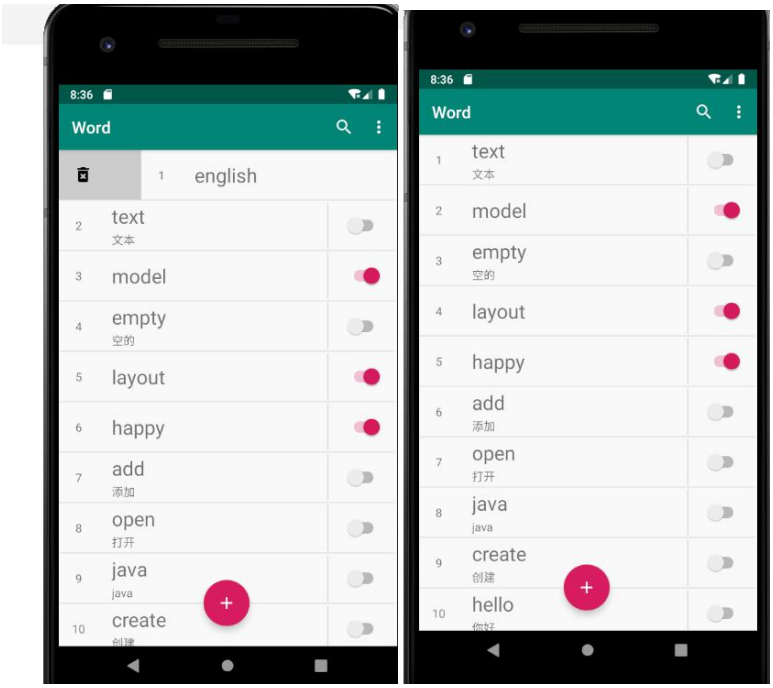
5.在部分手机版本中会出现搜索后单词丢失 **bug；**

二、运行过程示例

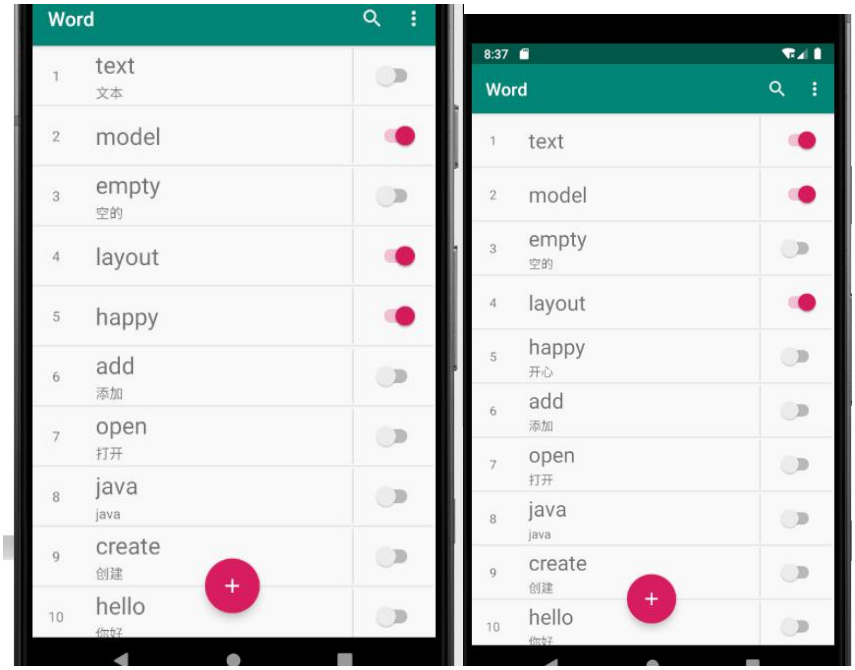
1. **添加单词：** ---单词添加成功



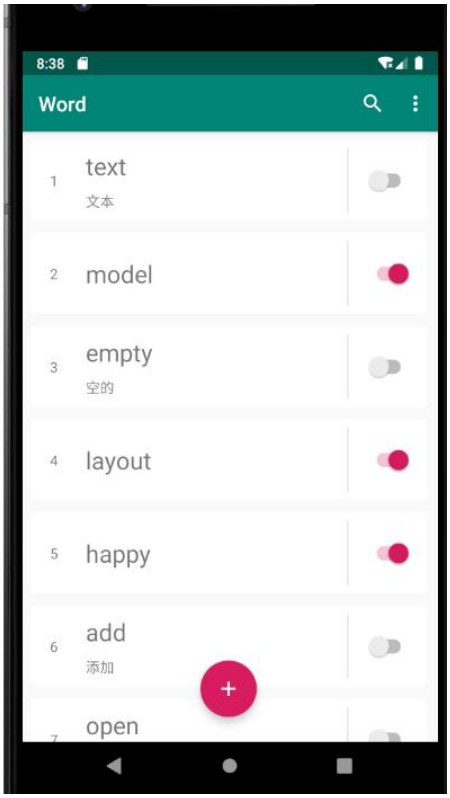
2. **删除单词：** ---删除单词成功



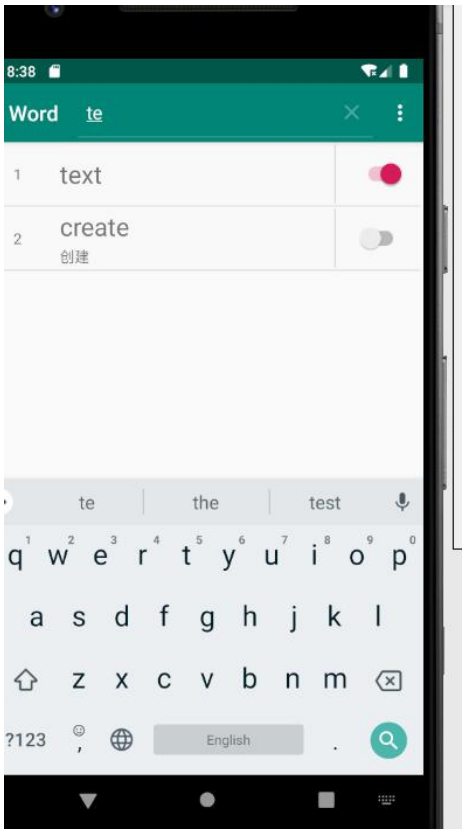
3. **中文释义显示/隐层：** -成功



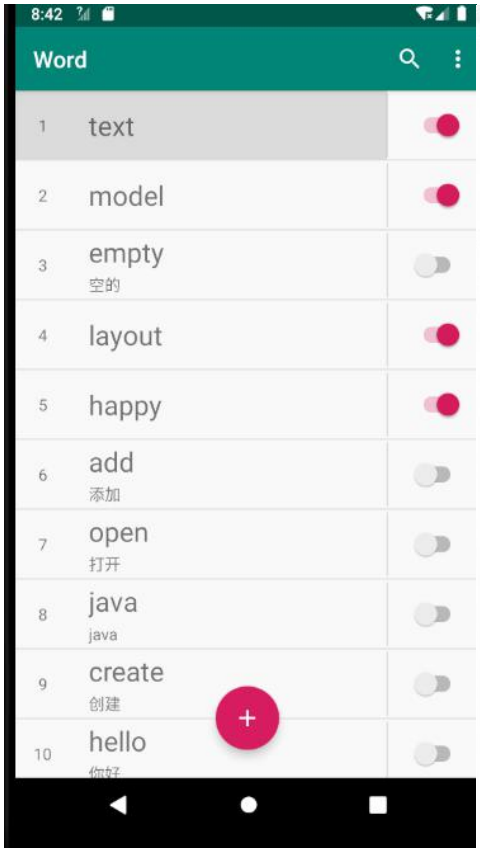
4. **切换视图：** -成功



5. **搜索功能：** --成功



6. **点击单词后跳到有道词典搜索详细详细** --成功



四、 个人小结

本课程学习了 andorid 的开发，经过本次项目，对 android 开发的整体把握更加清楚。
通过模块化开发，对每个类更加清楚，对 liveData..SharePreference..navigation 等类的使用和数据库的操作有了更加深入的理解。