一.搜索代码入口

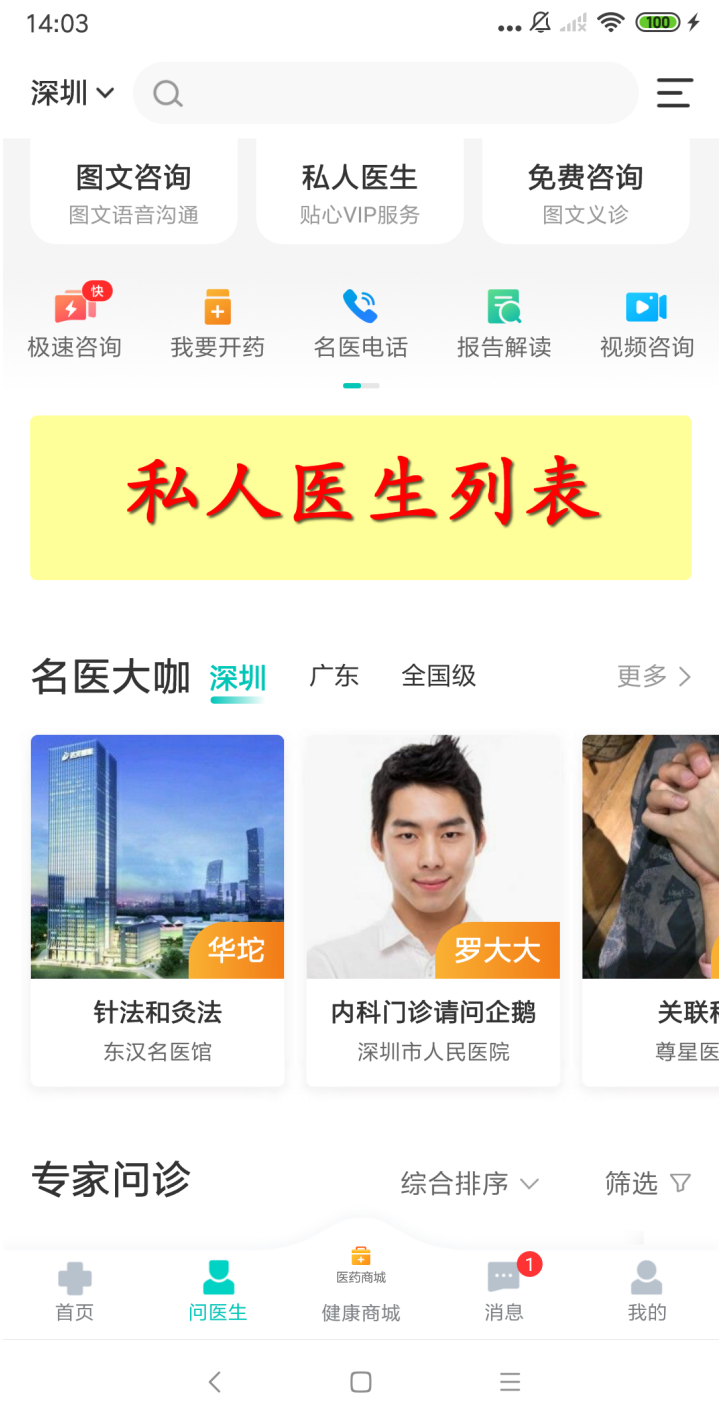
1.源码路径【cn.kidyn.qdmedical160.activity.search.searchnew/SearchIndexActivity】

2.其他跳转入口

```
* 入口：
* 1、问医生页 AskDoctorActivity
* 2、商城页 HealthActivity
* 3、挂号首页 HospitalListActivity
* 4、首页 HomeActivity
```

3.入口1AskDoctorActivity：

对应界面：



对应代码：AskDoctorActivity.java

```
 private void initSearch() {
        rlTopSearch.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                UmengMobclickAgentUntil.mobClickAgent(AskDoctorActivity.this,
EventIdObj.CONSULT_SEARCH_CLICK);

DataReportManager.getInstance().setFromTag(DataReportManager.FromConst.SECOND_TA
B);
                BiReportManager.getInstance().clickSubPage(new
SubPageEntity.Builder().setPageId(Page.WDO_SEO).build());

                SearchKeywordAd keyword = (SearchKeywordAd)
rlTopSearch.getTag();

ComprehensiveJumpUntil.goSearchIndexActivity(AskDoctorActivity.this,
SEARCH_ACTION.NORMAL, SEARCH_TAB.DOCTOR, null, keyword, null);
            }
        });
    }
```

二.搜索初始UI界面及逻辑

1.搜索入口界面

1.主要逻辑

1.onCreate()中调用init()

```
private void fetchData() {
        //热门搜索
        mViewModel.getHotKeywords(this);
        // 热门专题广告
        mViewModel.getHotSpecial(this);

        mViewModel.fetchAidWordList(this);
        mViewModel.fetchAllSearchAd(this);
    }
```

2.onResume()

```
@Override
   protected void onResume() {
       super.onResume();
       //获取历史搜索
       mViewModel.getSearchHistory(this);
       etSearch.postDelayed(() -> {
           if (null != mContext && null != etSearch) {
               ViewCommonUtils.showSystemInputNormal(mContext, etSearch);
           }
       }, 500);
   }
```

2.搜索联想界面

1.主要布局UI【activity_search_index.xml】

```
<!-- 搜索联想 -->
    <ListView
        android:id="@+id/list_search_associate"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@+id/layout_search"
        android:background="@color/white"
        android:divider="@null"
        android:listSelector="@null"
        android:visibility="gone" />
```

2.代码类

```
        associateAdapter = new SearchAssociateAdapter(this, associateList);
        associateAdapter.setColor(R.color.colorAccent);
        lvSearchAssociate.setAdapter(associateAdapter);
```

3.搜索结果入口

1.界面

2.跳转入口

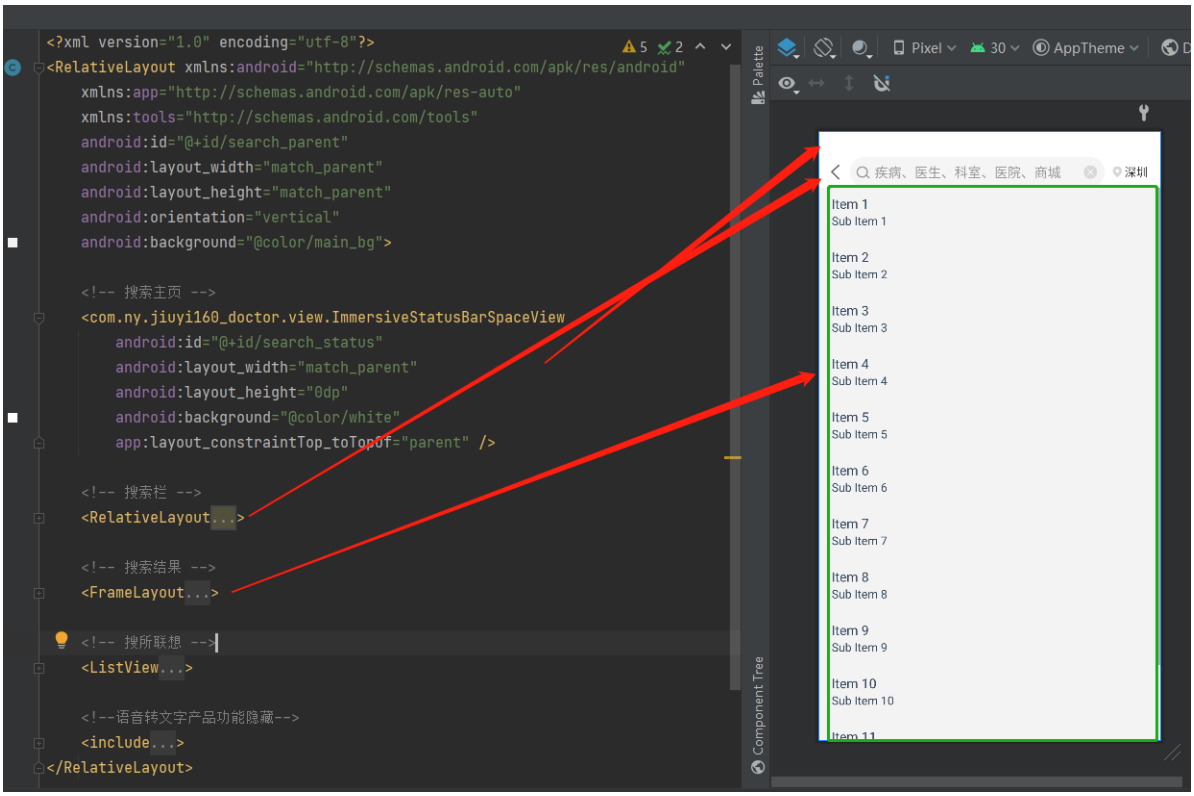
1.SearchIndexActivity#jumpToSearch()

```
 AidEntity aidEntity = mViewModel.containAidWords(keyword);
        if (aidEntity != null && !TextUtils.isEmpty(aidEntity.getWord_link())) {
            String url = ViewCommonUtils.appendCidForUrl(this,
aidEntity.getWord_link());
            ComprehensiveJumpUntil.gotoWebView2Activity(this, url, "");
            trackSearchEvent(keyword, keywordType);
        } else {
            ComprehensiveJumpUntil.gotoSearchNewActivity(this,
                    SEARCH_ACTION.AUTO_SEARCH,
                    mSearchTab,
                    null,
                    keywordType,
                    keyWordEntity, null, default_goods_type,
default_discount_type, null);
        }
    }
```

实际结果是到 SearchNewActivity.java类中【涉及界面较多，单独分析】

三.SearchNewActivity搜索结果页

1.整体界面结构



自上而下，分为状态栏，搜索栏，搜索结果，联想词【前面已说过，这里不再说明】

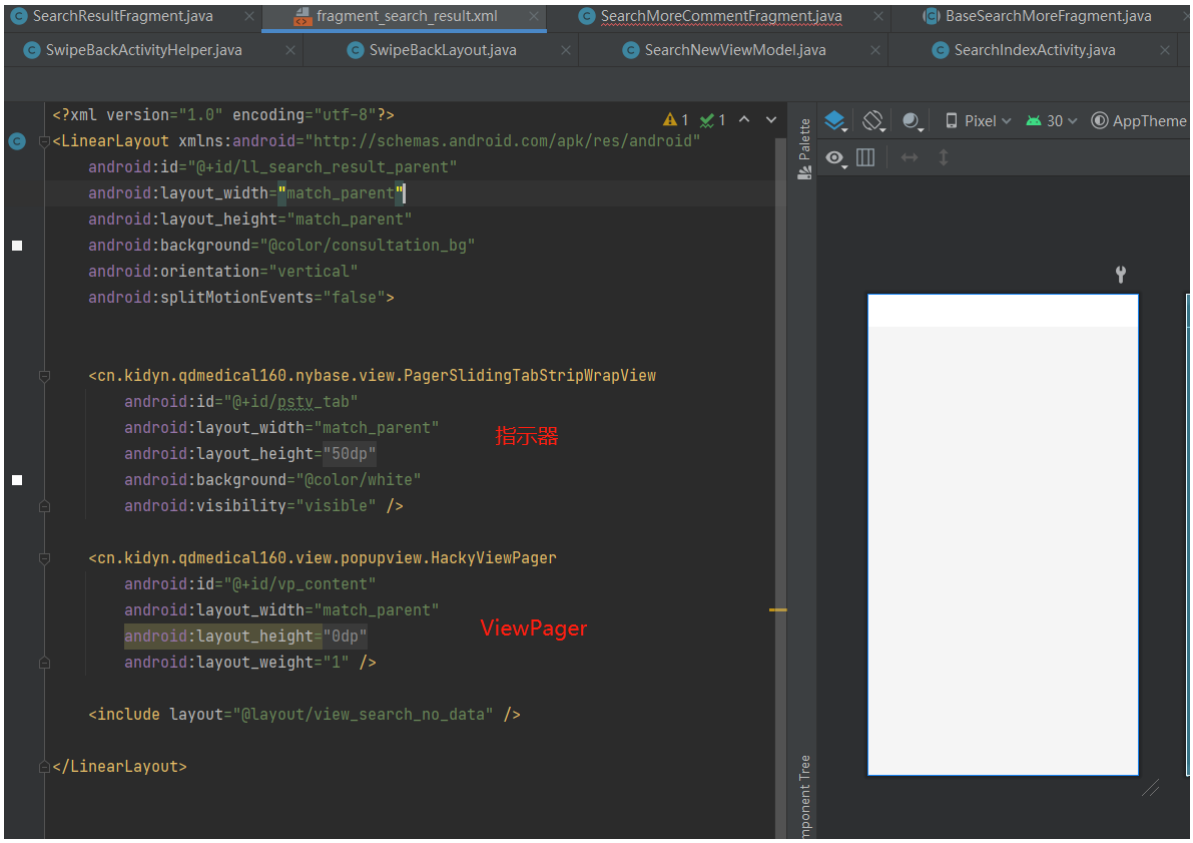
我们的重心主要是 下面每一项类别的数据结果，如下图红框位置，对应存放的是上图的蓝色框位置。

1.上图红色框部分对应的界面类是 SearchResultFragment.java,填充位置在

```
    private void initViews() {

        // 初始化搜索结果页

ViewCommonUtils.FragmentUtils.addFragmentWithoutRepeat(getSupportFragmentManager
(), R.id.search_result_layout, new SearchResultFragment());
       .....
       }
```

2.内容页SearchResultFragment.java

1.整体结构



2.填充数据过程

```
  private void concatData(List<SearchResultAll.TagList> tagLists, int index) {
        final List<Fragment> fragments = new ArrayList<>();
        List<String> titles = new ArrayList<>();
        for (SearchResultAll.TagList tag : tagLists) {
            titles.add(tag.getName());
            addFragments(fragments, tag.getType());
        }
        SearchPagerAdapter contentAdapter = new
SearchPagerAdapter(getChildFragmentManager(), fragments, titles);
        vpContent.setAdapter(contentAdapter);
        pstvTab.setViewPager(vpContent);

        SearchResultAll.TagList videoTag = new
SearchResultAll.TagList(SEARCH_TAB.VIDEO);
        pstvTab.setTabNew(tagLists.indexOf(videoTag), true);

    }

    /**
    * 添加Fragment进入列表
    *
    * @param fragments 需要放置Fragment的列表
    * @param type      需要放置的Fragment的类型
    */
   private void addFragments(List<Fragment> fragments, String type) {
        //curKeyword不能为空,否则会no-null导致崩溃
        if (curKeywords == null) {
            curKeywords = "";
        }
```

```
        if (keywords_type == null) {
            keywords_type = "";
        }
        if (SEARCH_TAB.ILL.equals(type)) {// 疾病
            fragments.add(SearchMoreIllFragment.newInstance(curKeywords,
from_unit_homepage, keywords_type));
        } else if (SEARCH_TAB.DOCTOR.equals(type)) { //医生
            fragments.add(SearchMoreDoctorFragment.newInstance(curKeywords,
from_unit_homepage, keywords_type));
        } else if (SEARCH_TAB.DEPARTMENT.equals(type)) {//科室
            fragments.add(SearchMoreDepartFragment.newInstance(curKeywords,
from_unit_homepage, keywords_type));
        } else if (SEARCH_TAB.HOSPITAL.equals(type)) {//医院
            fragments.add(SearchMoreHosFragment.newInstance(curKeywords,
from_unit_homepage, keywords_type));
        } else if (SEARCH_TAB.CONSULTATION.equals(type)) {//咨询
            fragments.add(SearchMoreConsultationFragment.newInstance(curKeywords,
from_unit_homepage, keywords_type));
        } else if (SEARCH_TAB.ALL.equals(type)) { //推荐
            fragments.add(SearchResultAllFragment.newInstance(curKeywords,
from_unit_homepage, keywords_type));
        } else if (SEARCH_TAB.ARTICLE.equals(type)) {//文章
            fragments.add(SearchMoreArticleFragment.newInstance(curKeywords,
from_unit_homepage, keywords_type));
        } else if (SEARCH_TAB.GOODS.equals(type)) {//商城
            fragments.add(SearchMoreGoodsFragment.newInstance(curKeywords,
from_unit_homepage, keywords_type,

getActivity().getIntent().getStringExtra(SearchIndexActivity.KEY_DEFAULT_GOODS_T
YPE),

getActivity().getIntent().getStringExtra(SearchIndexActivity.KEY_DEFAULT_DISCOUN
T_TYPE)));
        } else if (SEARCH_TAB.VIDEO.equals(type)) {//视频

fragments.add(SearchMoreVideoFragment.Companion.newInstance(curKeywords,
from_unit_homepage, keywords_type));
        } else if (SEARCH_TAB.COMMENT.equals(type)) {//咨询
            fragments.add(SearchMoreCommentFragment.newInstance(curKeywords,
from_unit_homepage, keywords_type));
        }
    }
```

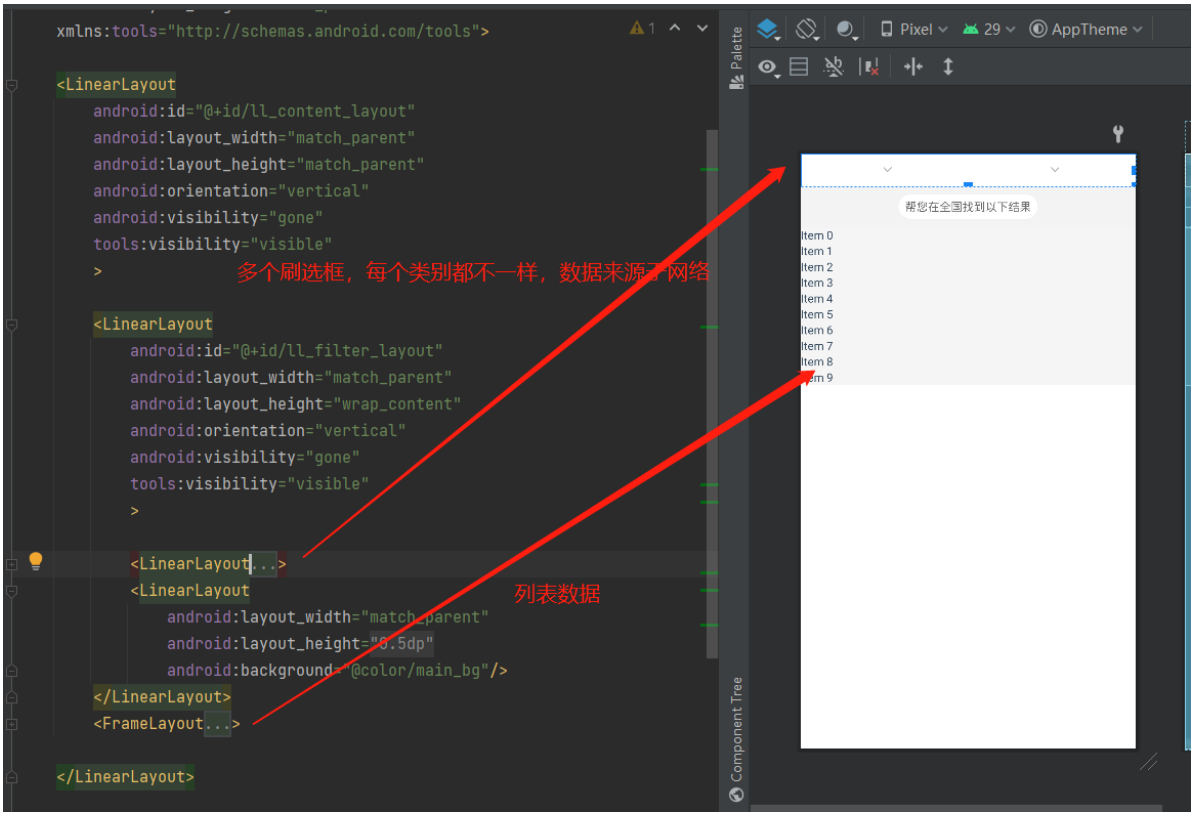
上面这几个item对应的Fragment主要的实现在于其父类 BaseSearchMoreFragment.java

3.BaseSearchMoreFragment

1.懒加载网络结果LazyLoadFragment

```
 * 1.使用setDelayTime(...)设置延迟时间
 * 2.子类覆写onCreateView(...)创建Root View（这一步和普通Fragment一样）
 * 3.子类实现loadData(...)加载或填充数据
```

2.界面结构



3.类别界面填充

```
 private void initView() {
        //注册不同类别对应的Item View
        register();
        //初始化刷选条件处理
        loadFilterView();
 }

  private void register() {
        // 这部分主要是为了设置 头部与尾部圆角背景的处理
        BaseSearchResultItemBinder.ItemBackGroundListener itemBackGroundListener
= new BaseSearchResultItemBinder.ItemBackGroundListener() {
            @Override
            public int positionOffset() {
                return 0;
            }

            @Override
            public boolean hasMore() {
                return hasMore;
            }

            @Override
            public int listSize() {
                return mqttPagerMultiTypeAdapter.getItemCount() - 1;
            }
```

```
            @Override
            public boolean hasSessionTitle() {
                if (SEARCH_TAB.DEPARTMENT.equals(mTabType)) {
                    return hasScheduleNowSortItem != null;
                }
                return false;
            }

            @Override
            public boolean hasSessionMore() {
                return false;
            }
        };
        // 医生
        SearchResultDoctorBinder searchResultDoctorBinder = new
SearchResultDoctorBinder(search_str);

searchResultDoctorBinder.setItemBackGroundListener(itemBackGroundListener);
        //采用MultiType库来设置不同的RecyclerView multiple view类型，数据类型通过
register（）第一个参数区别
        mqttPagerMultiTypeAdapter.register(SearchResultDoc.DocListEntity.class,
searchResultDoctorBinder);

        // 机构
        SearchResultUnitBinder searchResultUnitBinder = new
SearchResultUnitBinder(search_str);

searchResultUnitBinder.setItemBackGroundListener(itemBackGroundListener);
        mqttPagerMultiTypeAdapter.register(SearchResultUnit.UnitListEntity.class,
searchResultUnitBinder);

        // 科室
        SearchResultDepartmentBinder searchResultDepartmentBinder = new
SearchResultDepartmentBinder(search_str);

searchResultDepartmentBinder.setItemBackGroundListener(itemBackGroundListener);
        mqttPagerMultiTypeAdapter.register(SearchResultDep.DepListEntity.class,
searchResultDepartmentBinder);

        // 商品
        SearchResultGoodsBinder searchResultGoodsBinder = new
SearchResultGoodsBinder(search_str);

searchResultGoodsBinder.setItemBackGroundListener(itemBackGroundListener);

mqttPagerMultiTypeAdapter.register(SearchResultGoods.SearchGoodsEntity.class,
searchResultGoodsBinder);

        // 科谱
        SearchResultPublicScienceBinder searchResultPublicScienceBinder = new
SearchResultPublicScienceBinder(search_str);

searchResultPublicScienceBinder.setItemBackGroundListener(itemBackGroundListener
);

mqttPagerMultiTypeAdapter.register(SearchResultArticle.SearchArticleEntity.class,
searchResultPublicScienceBinder);
```

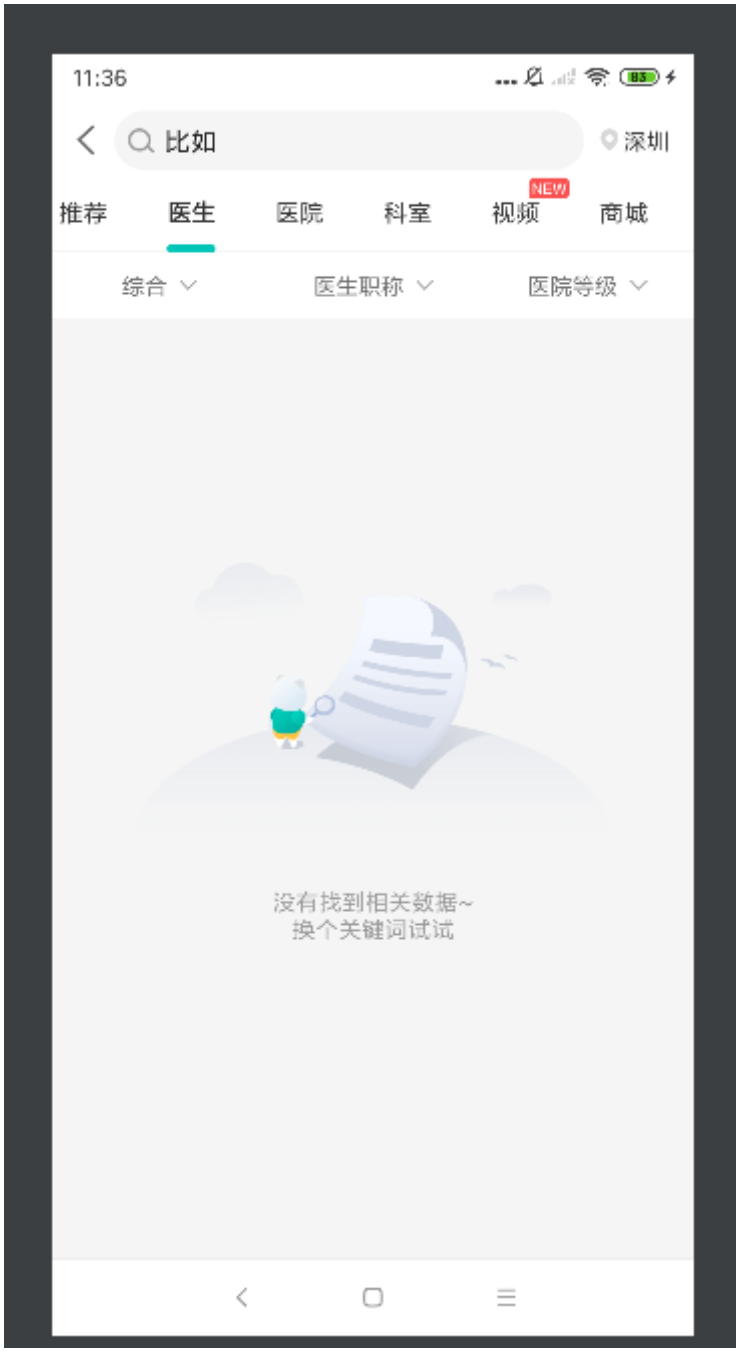
```
        // 咨询
        SearchResultConsultationBinder searchResultConsultationBinder = new
SearchResultConsultationBinder(search_str);

searchResultConsultationBinder.setItemBackGroundListener(itemBackGroundListener)
;
        mqttPagerMultiTypeAdapter.register(DocHomeConsultationItem.class,
searchResultConsultationBinder);

        // 点评
        SearchResultCommentBinder searchResultCommentBinder = new
SearchResultCommentBinder(search_str);

searchResultCommentBinder.setItemBackGroundListener(itemBackGroundListener);

mqttPagerMultiTypeAdapter.register(SearchResultComment.SearchCommentEntity.class,
searchResultCommentBinder);

    }
```

4.数据的加载与解析

在第一小节 “懒加载”中我们知道之类 重写 对应loadData(...) 函数 以及 BaseSearchMoreFragment#parseData()函数，即可做到不同类别数据的加载与 解析。

加载数据的请求最终到 BaseSearchMoreFragment#startReq()函数，根据不同tab类别设置不同的请求参数。

5.没有数据的展示情况



对应的控件是 <cn.kidyn.qdmedical160.nybase.view.StatusTipWidget>

但是 MqttPagerMultiTypeAdapter 已经做了没有数据的封装：

```
mqttPagerMultiTypeAdapter.setNoDataStatusTip(getString(R.string.text_search_no_r
esult));
mqttPagerMultiTypeAdapter.setNoDataStatus(R.drawable.bg_no_data_search);
```

四.需求-APP 搜索优化 - 医生/科室增加有号筛选

1.需求相关资料

tapd：https://www.tapd.cn/68476888/prong/tasks/view/1168476888001010314

UI：https://lanhuapp.com/web/#/item/project/stage?tid=574858b9-8e53-4bfa-96a0-711eb3c5901c&pid=baf35cc3-db18-490e-9934-b10b66d038eb

2.需求梳理

主要逻辑都在 BaseSearchMoreFragment.java类中，主要关注下面这个现有字段，理清思路会好一点。

```
/**
    * 医生tab和科室tab,展示是否有号排序头部view
    */
   protected AskDoctorOptionEntity.SortItem hasScheduleNowSortItem;
```

1.UI布局修改

在loadFilterView()初始化 过滤器的时候，增加对 医生，科室两个类别的 “优先展示当日有号的科室” 文案处理，并通过

```
protected void initFilterView(final String type) {
        mTabType = type;
        if (SEARCH_TAB.DOCTOR.equals(type)) {
            DoctorHasScheduleNowHeaderBinder doctorFilterBinder = new
DoctorHasScheduleNowHeaderBinder();
            doctorFilterBinder.setOnCheckedChangedListener(new Function1<Boolean,
Unit>() {
                @Override
                public Unit invoke(Boolean checked) {
                    if (hasScheduleNowSortItem != null) {
                        hasScheduleNowSortItem.setSelected(checked);
```

```
                        pullLayout.autoRefresh();
                    }
                    return null;
                }
            });

mqttPagerMultiTypeAdapter.register(AskDoctorOptionEntity.SortItem.class,
doctorFilterBinder);
        } else if (SEARCH_TAB.DEPARTMENT.equals(type)) {
            DepartmentHasScheduleNowHeaderBinder departmentFilterBinder = new
DepartmentHasScheduleNowHeaderBinder();
            departmentFilterBinder.setOnCheckedChangedListener(new
Function1<Boolean, Unit>() {
                @Override
                public Unit invoke(Boolean checked) {
                    if (hasScheduleNowSortItem != null) {
                        hasScheduleNowSortItem.setSelected(checked);
                        pullLayout.autoRefresh();
                    }
                    return null;
                }
            });

mqttPagerMultiTypeAdapter.register(AskDoctorOptionEntity.SortItem.class,
departmentFilterBinder);
        }
```

上面初始化布局后，数据在哪插入 getSearchSort()

```
getSeachSort(){
     if (!CollectionUtil.isEmpty(sortItems)) {
                        if (SEARCH_TAB.DOCTOR.equals(type) ||
SEARCH_TAB.DEPARTMENT.equals(type)) {
                            List<AskDoctorOptionEntity.SortItem> commonSortItems
= new ArrayList<>();
                            for (AskDoctorOptionEntity.SortItem item : sortItems)
{
                                if
("single_HasScheduleNow".equals(item.getType())) {
                                    hasScheduleNowSortItem = item;
                                } else {
                                    commonSortItems.add(item);
                                }
                            }
                            sortSortList = commonSortItems;
                        } else {
                            sortSortList = sortItems;
                        }
}
```

2.后端字段适配

1.标签 的控制展示字段

以医生类别对应的item布局 SearchResultDoctorBinder为例，“有号”标签 通过 hasScheduleNow字段展示，

```
  tvRegisterTag.visibility = if(data.hasScheduleNow == 1) View.VISIBLE else
View.GONE
```

hasScheduleNow是 “医生”类别数据，所以需要在 解析数据那里对于 这个字段进行赋值处理即可。

2.刷选条件字段的控制

主要关注 对于 hasScheduleNowSortItem的赋值

```
/**
     * 医生tab和科室tab,展示是否有号排序头部view
     */
    protected AskDoctorOptionEntity.SortItem hasScheduleNowSortItem;
```

五.推荐类别SearchResultAllFragment

1.整体界面结构

整体是一个滚动列表，不同的类别View

```
<FrameLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <com.nykj.pulllayout.PullLayout
        android:id="@+id/pull_layout"
        android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        android:splitMotionEvents="false"
        app:maxDownOffset="@dimen/pull_layout_header_height"
        app:maxRefreshTime="5000"
        app:maxUpOffset="@dimen/dimen_160dp"
        app:movieRatio="0.6"
        app:refreshOffset="@dimen/main_header_container_height">

        <androidx.recyclerview.widget.RecyclerView
            android:id="@+id/rv_result_list"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:animationCache="false"
            android:cacheColorHint="@color/transparent"
            android:divider="@null"
            android:dividerHeight="@dimen/dimen_0dp"
            android:overScrollMode="never"
            android:scrollingCache="false"
            />

        <cn.kidyn.qdmedical160.view.listview.PullHeaderView
            android:layout_width="match_parent"
            android:layout_height="@dimen/pull_layout_header_height"
            android:background="#F5F5F5" />
    </com.nykj.pulllayout.PullLayout>
</FrameLayout>
```


9:59
张大哥
深圳
推荐
医生
医院
科室
视频
NEW
商城
诊前咨询 快
怕挂错号？不知道病情轻重？
1分钟快速找到医生，6000+医生在线，320万人次已咨询
¥5 起
医生
张大哥 院长 研究员 主任医师
曾就职西乡医院2 | 总店-消化内科
9.2分 服务 3949 回复速度 大于4小时
名医大咖专病榜第1名
感冒
病态窦房结综合征
耳石症
焦虑症
尿失禁
图文咨询
6元/次
视频咨询
10元/15分钟
电话咨询
10元/10分钟
医院
张哥中医理疗馆
8.2分 综合医院 三甲
5.0m
东滨路4078号永新汇2栋9楼、...
南山粤海
服务商品+物流 8.8折 ¥777
161-48单人贵1宾体检套... 0.1折 ¥99

可以看到 自上而下有一个

2.数据填充【参考】

整体采取 MqttPagerMultiTypeAdapter 方式，利用不同的数据类型添加不同的item布局，比如

```
// 分割线
mqttPagerMultiTypeAdapter.register(SearchResultSessionDivider.Type.class, new
SearchResultSessionDivider());
类别-【医生,科室】
mqttPagerMultiTypeAdapter.register(SearchResultSessionHeaderBinder.HeaderEntity.
class, new SearchResultSessionHeaderBinder());
// 底部查看更多
SearchResultSessionFooterBinder searchResultSessionFooterBinder = new
SearchResultSessionFooterBinder();
// 具体某一个类别
// 广告
mqttPagerMultiTypeAdapter.register(SearchResultAdvertisement.class, new
SearchResultAdvertisementBinder());
//诊前广告
mqttPagerMultiTypeAdapter.register(Banner.class, new
SearchResultBannerAdvertisementBinder());
 // 疾病
mqttPagerMultiTypeAdapter.register(SearchResultIll.class, new
SearchResultDiseaseBinder(curKeywords));
```

```
mqttPagerMultiTypeAdapter.register(SearchResultRelatedDiseaseBinder.RelatedDisea
se.class, new SearchResultRelatedDiseaseBinder());

// 机构
SearchResultUnitBinder searchResultUnitBinder = new
SearchResultUnitBinder(curKeywords);
searchResultUnitBinder.setItemBackGroundListener(getItemBackgroundListener(SEARC
H_TAB.HOSPITAL));
mqttPagerMultiTypeAdapter.register(SearchResultUnit.UnitListEntity.class,
searchResultUnitBinder);

// 医生
SearchResultDoctorBinder searchResultDoctorBinder = new
SearchResultDoctorBinder(curKeywords);
searchResultDoctorBinder.setItemBackGroundListener(getItemBackgroundListener(SEA
RCH_TAB.DOCTOR));
mqttPagerMultiTypeAdapter.register(SearchResultDoc.DocListEntity.class,
searchResultDoctorBinder);

// 科室
SearchResultDepartmentBinder searchResultDepartmentBinder = new
SearchResultDepartmentBinder(curKeywords);

searchResultDepartmentBinder.setItemBackGroundListener(getItemBackgroundListener
(SEARCH_TAB.DEPARTMENT));
mqttPagerMultiTypeAdapter.register(SearchResultDep.DepListEntity.class,
searchResultDepartmentBinder);

// 笔记
        SearchResultAllNoteBinder searchResultAllNoteBinder = new
SearchResultAllNoteBinder(curKeywords);
        searchResultAllNoteBinder.setOnMoreClickListener(() -> {
            moreClick(SEARCH_TAB.VIDEO);
        });
        mqttPagerMultiTypeAdapter.register(SearchResultNote.class,
searchResultAllNoteBinder);

        // 科谱
        SearchResultPublicScienceBinder searchResultPublicScienceBinder = new
SearchResultPublicScienceBinder(curKeywords);

searchResultPublicScienceBinder.setItemBackGroundListener(getItemBackgroundListe
ner(SEARCH_TAB.ARTICLE));

mqttPagerMultiTypeAdapter.register(SearchResultArticle.SearchArticleEntity.class,
searchResultPublicScienceBinder);

        // 商品
        SearchResultGoodsBinder searchResultGoodsBinder = new
SearchResultGoodsBinder(curKeywords);

searchResultGoodsBinder.setItemBackGroundListener(getItemBackgroundListener(SEAR
CH_TAB.GOODS));

mqttPagerMultiTypeAdapter.register(SearchResultGoods.SearchGoodsEntity.class,
searchResultGoodsBinder);

        // 问答
```

```
        SearchResultConsultationBinder searchResultConsultationBinder = new
SearchResultConsultationBinder(curKeywords);

searchResultConsultationBinder.setItemBackGroundListener(getItemBackgroundListen
er(SEARCH_TAB.CONSULTATION));
        mqttPagerMultiTypeAdapter.register(DocHomeConsultationItem.class,
searchResultConsultationBinder);

        // 点评
        SearchResultCommentBinder searchResultCommentBinder = new
SearchResultCommentBinder(curKeywords);

searchResultCommentBinder.setItemBackGroundListener(getItemBackgroundListener(SE
ARCH_TAB.COMMENT));

mqttPagerMultiTypeAdapter.register(SearchResultComment.SearchCommentEntity.class,
searchResultCommentBinder);
```

六.商城SearchMoreGoodsFragment

1.UI结构


```
public class SearchMoreGoodsFragment extends
BaseSearchMoreFragment<SearchResultGoods.SearchGoodsEntity> {}
```

继承于 `BaseSearchMoreFragment`，所以相关实现都在这里面

2.实现分类别展示【参考】

对应的itemView是：

```
BaseSearchMoreFragment：
// 商品
SearchResultGoodsBinder searchResultGoodsBinder = new
SearchResultGoodsBinder(search_str);
searchResultGoodsBinder.setItemBackGroundListener(itemBackGroundListener);
mqttPagerMultiTypeAdapter.register(SearchResultGoods.SearchGoodsEntity.class,
searchResultGoodsBinder);
```

需要注意的是 这里设置了一个 itemBackGroundListener：

```
private void register() {
    BaseSearchResultItemBinder.ItemBackGroundListener itemBackGroundListener =
new BaseSearchResultItemBinder.ItemBackGroundListener() {
        @Override
        public int positionOffset() {
            return 0;
        }

        @Override
        public boolean hasMore() {
            return hasMore;
        }

        @Override
        public int listSize() {
            return mqttPagerMultiTypeAdapter.getItemCount() - 1;
        }

        @Override
        public boolean hasSessionTitle() {
            if (SEARCH_TAB.DEPARTMENT.equals(mTabType)) {
                return hasScheduleNowSortItem != null;
            }
            if (SEARCH_TAB.GOODS.equals(mTabType)) {
                return hasRecommend != null;
            }
            return false;
        }

        @Override
        public boolean hasSessionMore() {
            return false;
        }
    };
```

这里的用法主要用于“根据item处的位置设置设置背景 ”，这块比较重要，

比如我们想实现下面的效果：【搜索推荐SearchResultAllFragment 】

分类别 隔开，同类别的 效果看起来在同一个 背景圆角下。

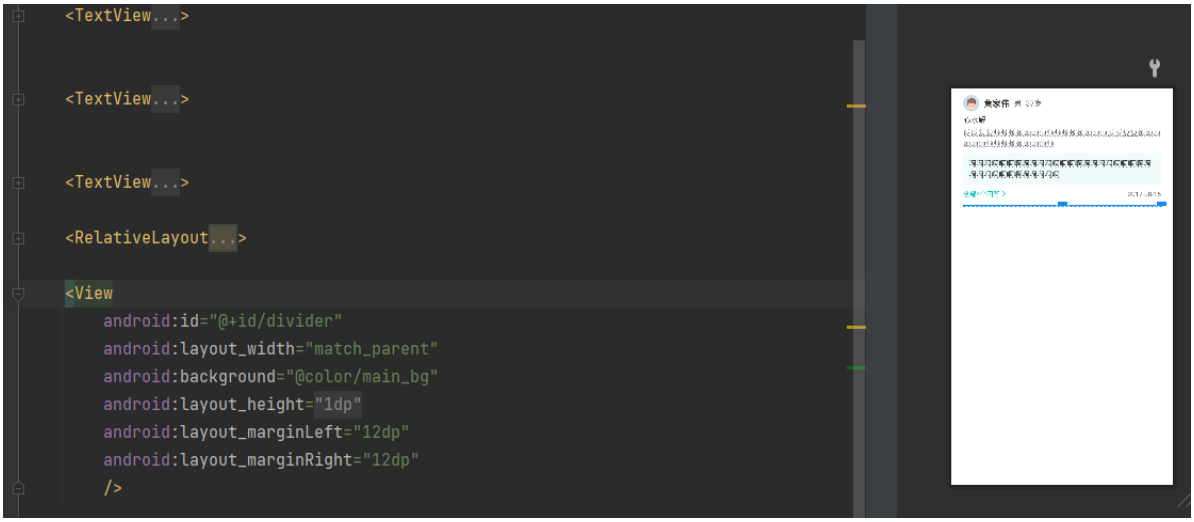
那么是怎么实现的呢，实际上 “商场”，以及下面的 “张哥xxx” 都对应一个单独的ItemView，下面是思路

```
1.设置recycleView的间隔为0
recyclerView.setLayoutManager(new LinearLayoutManager(activity));
recyclerView.setItemAnimator(null);
recyclerView.setAdapter(mqttPagerMultiTypeAdapter);
SpacingDecoration spacingDecoration = new SpacingDecoration(activity, 0);
spacingDecoration.setOutSpacing(activity, 0, 0, 0, 40);
recyclerView.addItemDecoration(spacingDecoration);

2.设置不同的itemView
//中间的间隔，比如上图 科室类别与商场类别中间的 空格
mqttPagerMultiTypeAdapter.register(SearchResultSessionDivider.Type.class, new
SearchResultSessionDivider());

mqttPagerMultiTypeAdapter.register(SearchResultSessionHeaderBinder.HeaderEntity.
class, new SearchResultSessionHeaderBinder());

注册数据类别：
// 科室
        SearchResultDepartmentBinder searchResultDepartmentBinder = new
SearchResultDepartmentBinder(curKeywords);
```

```
searchResultDepartmentBinder.setItemBackGroundListener(getItemBackgroundListener
(SEARCH_TAB.DEPARTMENT));
        mqttPagerMultiTypeAdapter.register(SearchResultDep.DepListEntity.class,
searchResultDepartmentBinder);

    // 商品
        SearchResultGoodsBinder searchResultGoodsBinder = new
SearchResultGoodsBinder(curKeywords);

searchResultGoodsBinder.setItemBackGroundListener(getItemBackgroundListener(SEAR
CH_TAB.GOODS));

mqttPagerMultiTypeAdapter.register(SearchResultGoods.SearchGoodsEntity.class,
searchResultGoodsBinder);
```

3.设置 setItemBackGroundListener

4.类别相同的，需要自己实现一个 横线

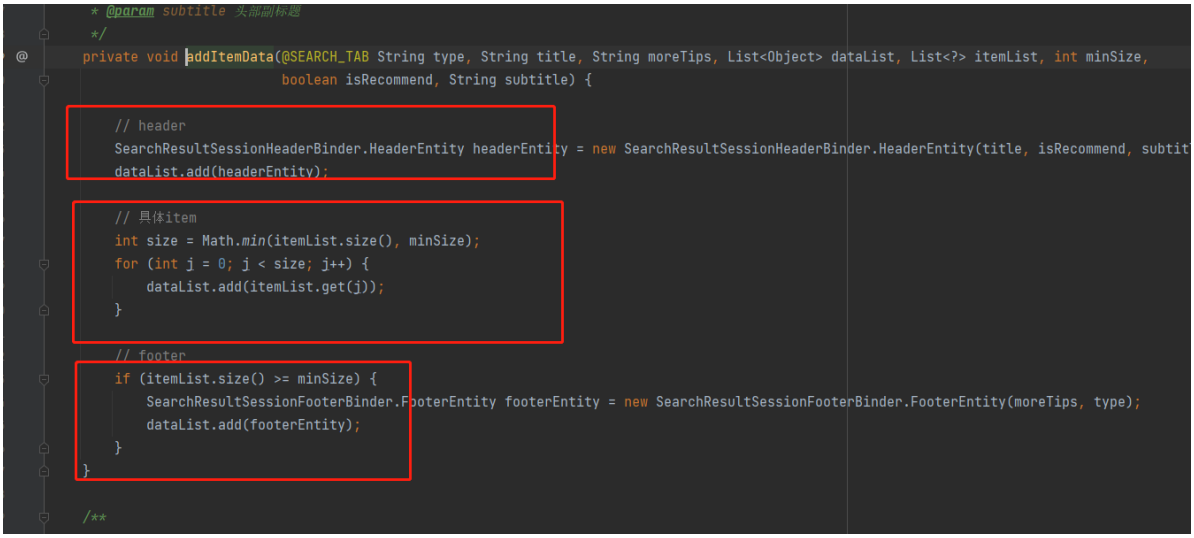


5.在不同的itemView绑定数据的时候调用 updateItemViewBackground(itemView,binding.divider,itemPosition)

```
inner class ViewHolder(itemView: View):RecyclerView.ViewHolder(itemView){
    private val binding by
viewBinding(ItemSearchResultConsultationBinding::bind)
    private val mContext = itemView.context

    fun bind(consultationItem: DocHomeConsultationItem){

        val itemPosition = getPosition(this)
        updateItemViewBackground(itemView,binding.divider,itemPosition)
        biReport(
            Page.Se00_004,
            "",
            "",
            BhvType.SEARCH_EXPOSE,
            curKeywords,
            consultationItem.report_data,
            itemPosition.toString()
        )

          /**
     * 根据item处的位置设置设置背景
     * 包括了在推荐页和各自的搜索结果页的情况
     */
    protected fun updateItemViewBackground(itemView:
View,divider:View?,itemPosition:Int){

        itemBackGroundListener?.let {
            val realPosition = itemPosition - it.positionOffset()
            if(it.listSize() == 1){
                if(it.hasSessionTitle()){

itemView.setBackgroundResource(R.drawable.shape_search_result_corner_bottom_bg)
                }else{

itemView.setBackgroundResource(R.drawable.shape_search_result_doctor_bg)
                }
                divider?.visibility = View.GONE
            }else{
                val bgRes: Int = when (realPosition) {
                    0 -> {
                        divider?.visibility = View.VISIBLE
                        if (it.hasSessionTitle()) {
                            R.drawable.shape_search_result_corner_0dp_bg
                        } else {
                            R.drawable.shape_search_result_corner_top_bg
                        }
                    }
                    it.listSize() - 1 -> {
                        if (it.hasSessionMore()) {
                            divider?.visibility = View.VISIBLE
                            R.drawable.shape_search_result_corner_0dp_bg
                        } else {
                            if (it.hasMore()) {
                                divider?.visibility = View.VISIBLE
                                R.drawable.shape_search_result_corner_0dp_bg
                            } else {
                                divider?.visibility = View.GONE
                                R.drawable.shape_search_result_corner_bottom_bg
```

```
                            }
                        }
                    }
                    else -> {
                        divider?.visibility = View.VISIBLE
                        R.drawable.shape_search_result_corner_0dp_bg
                    }
                }
                itemView.setBackgroundResource(bgRes)
            }
        }
    }
```

6.自己向adapter插入数据



```
     * @param subtitle 头部副标题
     */
@    private void addItemData(@SEARCH_TAB String type, String title, String moreTips, List<Object> dataList, List<?> itemList, int minSize,
                              boolean isRecommend, String subtitle) {

        // header
        SearchResultSessionHeaderBinder.HeaderEntity headerEntity = new SearchResultSessionHeaderBinder.HeaderEntity(title, isRecommend, subtit
        dataList.add(headerEntity);

        // 具体item
        int size = Math.min(itemList.size(), minSize);
        for (int j = 0; j < size; j++) {
            dataList.add(itemList.get(j));
        }

        // footer
        if (itemList.size() >= minSize) {
            SearchResultSessionFooterBinder.FooterEntity footerEntity = new SearchResultSessionFooterBinder.FooterEntity(moreTips, type);
            dataList.add(footerEntity);
        }
    }

    /**
```

这样子就能够实现 一个RecycleView，多种不同类别的ItemView展示。

这种实现方式 在**搜索模块**，**机构主页：HospitalHomeFragment** 都很经常使用。