Android软件设计课程实验

Android软件设计课程实验报告

学号 SA20225506

姓名 谢雪松

日期 2020.10.27

|  |  |
| --- | --- |
| **实验名称** | 实验二：实现Activity跳转以及控件的综合使用 |
| **实验内容** | 要求：1. Button跳转页面；2. ListView使用；3. ImageView使用；4. 布局综合使用；  评分：1. 完成上述4项基本UI设计和功能（7分）；  2. 可根据UI美化效果适当加分或模仿微信或手机QQ的UI和框架设计； |
| **实验完成情况**  （包括完成的实验内容及每个实验的完成程度。注意要贴出每个实验的  核心代码） | 完成的实验内容   1. 完成了Button跳转页面。 核心代码：   login\_bt.setOnClickListener(this); @Override public void onClick(View v) {  switch (v.getId()) {  case R.id.loginbutton:  //执行登录操作  login\_action();  break;  …………………….  } } //登录操作 private void login\_action(){  ………….  Intent intent = new Intent(LoginActivity.this, HomeActivity.class );  intent.putExtra("name",name);  startActivity(intent);  ……… }   1. 使用了ListView，配合BaseAdapter展示用户安装的应用   核心代码  public class AllListAdapter extends BaseAdapter implements Filterable {  ... } appAdapter =new AllListAdapter(getContext()); for(PackageInfo packageInfo:packages){  app.set...(...);  appAdapter.add(app); } app\_list\_view.setAdapter(appAdapter);   1. 多次使用到了ImageView，进行用户头像、默认提示图片、应用图标的显示。   Drawable avatar\_drawable = db.getAvatarByName(name); if(avatar\_drawable==null){  avatar\_img.setImageResource(R.drawable.*boy*); } else {  avatar\_img.setImageDrawable(avatar\_drawable); }   1. 通过RadioButton、Fragment配合，使得HomeActivity内通过底部导航按钮切换不同的Fragment。   @Override public void onCheckedChanged(RadioGroup group, int checkedId) {  transaction=fm.beginTransaction();  switch (checkedId){  case R.id.*radio1*:  fragment=fragments.get(0);  transaction.replace(R.id.*mFragment*,fragment,"all");  break;  case R.id.*radio2*:  fragment=fragments.get(1);  transaction.replace(R.id.*mFragment*,fragment,"favorite");  break;  }  transaction.commit(); }   1. 使用到了RecyclerView配合RecyclerView.Adapter来显示用户的应用收藏列表。   favoriteListview =getActivity().findViewById(R.id.*users\_listview*); mAdapter=new FavoriteListAdapter(getContext(),db\_name); favoriteListview.setLayoutManager(new GridLayoutManager(getContext(),2)); favoriteListview.setAdapter(mAdapter);   1. 使用了Sqlite数据库完成对用户登录信息的持久化操作。   User类保存一个用户的用户名、密码等信息，对应数据库中的一条记录。  UsersDBHelper类继承自SQLiteOpenHelper，完成了数据库创建、表创建、插入用户记录、查询用户等操作。  7、通过Google官方提供的Room库来进行SQLite数据库操作，实现了对用户收藏列表的持久化操作。  AppBean类注解为room中的Entity，AppDAO接口对应room中的DAO，AppDataBase抽象类对应room中的database。在使用room的过程中，不需要自己手写具体实现，而是通过注解，room框架会自动帮我们实现，大大简化了数据库的开发。 |
| **实验中的问题**  （包括在实验中遇到的问题，以及解决问题的方法） | 1. 如何通过数据库保存和读取用户的头像。   在建表时，声明头像（avatar）为blob类型的字段  final String CREATE\_USER="create table User ("  + "uid integer primary key autoincrement,"  + "name text ,"  + "password text,"  + "email text,"  + "avatar blob)"; db.execSQL(CREATE\_USER);  头像从数据库中读取为byte数组，可以将其转换为Drawable对象显示在界面上。  if((bytes=cursor.getBlob(cursor.getColumnIndex("avatar")))!=null){  avatar= new BitmapDrawable(BitmapFactory.*decodeByteArray*(bytes, 0, bytes.length)); }  同理，将图片转化为byte数组后，可以存如数据库。  public byte[] getAvatarBytes(){  Bitmap imageBitmap = ((BitmapDrawable) d).getBitmap();  ByteArrayOutputStream baos = new ByteArrayOutputStream();  imageBitmap.compress(Bitmap.CompressFormat.PNG, 100, baos);  byte[] byteData = baos.toByteArray(); } contentValues.put("avatar", User.getAvatarBytes()); db.insert(TABLE\_NAME,null,contentValues);   1. 如何对所有应用列表进行搜索，方便用户查找到含有关键词的应用。   给搜索框添加文本改变时的监听器，当文本改变时，会通知listview的适配器过滤得到含有关键字的app的名称。  public class AllAppsFragment extends Fragment implements View.OnClickListener, TextWatcher {  search\_box.addTextChangedListener(this);  @Override  public void onTextChanged(CharSequence s, int start, int before, int count) {  appAdapter.getFilter().filter(s);  } } public class AllListAdapter extends BaseAdapter implements Filterable {  @Override  public Filter getFilter() {  ...  if (app.getAppname().toLowerCase().contains(constraint.toString())) {  filteredApps.add(app);  }  ...  } }   1. 如何让不同用户有不同的收藏列表。   获得登录界面传来的用户名信息，在创建数据库访问对象时，数据库名字包含用户名信息，即可让不同用户有不同的收藏列表。  Intent intent=getIntent(); name=intent.getStringExtra("name"); String db\_name=userName+".db"; AppDataBase db = Room.*databaseBuilder*(context, AppDataBase.class, db\_name).allowMainThreadQueries().build();   1. 不同Fragment之间如何调用对方的函数。   先在一个Fragment里声明一个public的方法  public void insertApp(AppBean app){  mAdapter.insertApp(app); }  在HomeActivity切换不同Fragment时指明一个标签  transaction.replace(R.id.*mFragment*,fragment,"favorite");  在另一个Fragment里通过标签获取之前fragment的引用  favoriteFragment=(FavoriteFragment) fragmentManager.findFragmentByTag("favorite");  之后就可以方便的调用之前Fragment的方法了  favoriteFragment.insertApp(app);  5、如何让用户的收藏列表可以显示为网格状。  可以用多个listview来实现，但这样效率不高。通过查找资料，谷歌官方推荐使用RecyclerView来实现列表的网格视图。  favoriteListview.setLayoutManager(new GridLayoutManager(getContext(),2));  如上代码，可以实现列表显示为网格视图，每一行有两格。   1. 如何在项目中使用room框架   在项目或模块的build.gradle文件里添加依赖：  dependencies **{  . . . . .**   //room库的依赖  def room\_version = "2.2.5"  implementation "androidx.room:room-runtime:$room\_version"  annotationProcessor "androidx.room:room-compiler:$room\_version"  // optional - RxJava support for Room  implementation "androidx.room:room-rxjava2:$room\_version"  // optional - Guava support for Room, including Optional and ListenableFuture  implementation "androidx.room:room-guava:$room\_version"  // optional - Test helpers  testImplementation "androidx.room:room-testing:$room\_version" **}** |
| **实验结果**  **及成果展示**  （包括实验完成后的源码和打包文件的说明） | 一、成果展示和完成后的源码  1. 登录界面     这是登录界面，头像处使用ImageView实现，当用户输入用户名后，会从数据库找到对应的用户头像，并显示出来,如果没有该用户，或该用户没有头像，则显示默认头像图片。  用户名和密码输入框使用EditText实现，登录按钮使用了Button。用户名密码匹配成功则会跳转到主界面。下面忘记密码和用户注册使用的是TextView，点按会跳转到相应的界面。   1. 注册界面源码 点击用户注册会跳转到该界面。   输入用户名密码、选择头像，会在数据库中创建一个新的用户。     1. 重设密码界面   点击忘记密码TextView会跳转到这个页面。  在输入框中输入用户名和注册所用的邮箱，如果匹配则可以成功修改密码。有相应的toast提醒修改成功、用户不存在、信息不匹配等情况。   1. 主界面   登录成功后会进入主界面。 主界面由我的收藏和所有应用两个板块组成，通过下方的按钮可以实现切换。     1. 所有应用Fragment   在所有应用fragment获取并显示手机的所有应用，点击左侧的星星按钮可以把应用添加到用户的收藏列表。    加了星的应用会在我的收藏页面中显示。    点击右上角的搜索按钮，会显示出搜索框。输入文字过程中，会实时更新和输入文字相匹配的应用。   1. 我的收藏Fragment 点击下方我的收藏可进入我的收藏页面。   这里显示了用户加星收藏的应用。点击应用图标会显示两个操作按钮。      点击右边的“垃圾桶”图标可将对应的应用从收藏列表中删除。（右图为删除“有道云笔记”后的界面）  点击左边的“播放”图标可以打开对应的应用。第一次打开时，系统会提示，点击“允许”后将会打开应用“百度”，这里不进行显示。   二、打包文件说明 app2/ 项目名称  ├── build.gradle gradle的配置文件  ├── gradle  │   └── wrapper  ├── gradle.properties  ├── gradlew  ├── gradlew.bat  ├── local.properties  ├── settings.gradle  └── xxs 模块名称  ├── build  ├── build.gradle  ├── proguard-rules.pro  └─ src  └── main  ├── AndroidManifest.xml 应用清单  ├── java  │   └── com  │   └── rom471  │   ├── adapters  │   │   ├── AllListAdapter.java 所有应用列表的适配器  │   │   └── FavoriteListAdapter.java 收藏应用列表  │   ├── appsdb  │   │   ├── AppBean.java 收藏应用记录类  │   │   ├── AppDAO.java 收藏应用的数据库操作接口  │   │   └── AppDataBase.java 收藏应用的数据库抽象类  │   ├── main  │   │   ├── HomeActivity.java 主界面  │   │   ├── LoginActivity.java 登录界面  │   │   ├── RegisterActivity.java 注册界面  │   │   ├── ResetPasswordActivity.java 重设密码界面  │   │   └── fragments  │   │   ├── AllAppsFragment.java 所有应用的fragment  │   │   └── FavoriteFragment.java 收藏应用的fragment  │   └── userdb  │   ├── MyUtils.java 用户数据库操作的工具类  │   ├── User.java 用户记录类  │   └── UsersDBHelper.java 用户数据库操作类  └── res  ├── anim  │   └── item\_add\_anim.xml 所有应用界面点击item后的动画  ├── drawable  │   ├── apps.png  │   ├── apps\_fill.png  │   ├── apps\_selector.xml 底栏“所有应用”图标的选择器  │   ├── avatar3.jpg 初始用户wkq的头像  │   ├── avatar\_wjf.jpg 初始用户wjf的头像  │   ├── avatar\_xxs.jpg 初始用户xxs的头像  │   ├── boy.png 用户提示头像  │   ├── btn\_round.xml  │   ├── button\_selector.xml  │   ├── changepassword.png  │   ├── default\_avatar.png  │   ├── delete.png  │   ├── edittext\_selector.xml  │   ├── find.png  │   ├── ic\_launcher\_background.xml  │   ├── like.png  │   ├── like\_fill.png  │   ├── like\_selector.xml 底栏“收藏应用”图标的选择器  │   ├── open.png  │   ├── star.png  │   └── stared.png  ├── layout  │   ├── activity\_home.xml 主界面布局  │   ├── activity\_home\_buttom.xml 主界面底栏布局  │   ├── activity\_register.xml 注册界面布局  │   ├── activity\_reset\_password.xml 重设密码界面布局  │   ├── app\_list\_layout.xml 所有应用列表布局  │   ├── favorite\_app\_item.xml 收藏应用列表布局  │   ├── favorite\_app\_item\_menu.xml 收藏应用点击后打开的菜单布局  │   ├── layout\_all\_fragment.xml 所有应用的fragment布局  │   ├── layout\_favorite\_fragment.xml 收藏应用的fragment布局  │   └── login\_layout.xml 登录界面的布局  └── values  ├── colors.xml 一些颜色常量  ├── strings.xml 一些字符串常量  └── styles.xml 一些样式常量 三、实验源码 LoginActivity.java package com.rom471.main; import android.content.Intent; import android.graphics.drawable.Drawable; import android.os.Bundle; import android.text.Editable; import android.text.TextWatcher; import android.view.Gravity; import android.view.View; import android.widget.Button; import android.widget.EditText; import android.widget.ImageView; import android.widget.TextView; import android.widget.Toast;  import androidx.appcompat.app.AppCompatActivity;  import com.rom471.lab1.R; import com.rom471.userdb.UsersDBHelper;  public class LoginActivity extends AppCompatActivity implements View.OnClickListener {  EditText name\_et; //用户名输入框  EditText password\_et;//用户密码输入框  UsersDBHelper db; //用户数据库操作类  ImageView avatar\_img; //用户头像图片  TextView register\_tv; //注册按钮  TextView reset\_tv; //重设密码按钮   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*login\_layout*);  //绑定资源  name\_et=findViewById(R.id.*Acount*);  avatar\_img=findViewById(R.id.*pic*);  password\_et=findViewById(R.id.*passwd*);  Button login\_bt=findViewById(R.id.*loginbutton*);  register\_tv=findViewById(R.id.*register\_tv*);  reset\_tv=findViewById(R.id.*forget*);   //设置按钮的点击事件监听器  login\_bt.setOnClickListener(this);  register\_tv.setOnClickListener(this);  reset\_tv.setOnClickListener(this);  //根据名字输入框动态的查找用户的头像  name\_et.addTextChangedListener(new TextWatcher() {  @Override  public void beforeTextChanged(CharSequence s, int start, int count, int after) { }  @Override  public void onTextChanged(CharSequence s, int start, int before, int count) {  String name=s.toString();//获取用户输入的用户名  if(!name.equals("")){  Drawable avatar\_drawable = db.getAvatarByName(name);  if(avatar\_drawable==null){  avatar\_img.setImageResource(R.drawable.*boy*);  }  else {  avatar\_img.setImageDrawable(avatar\_drawable);  }  }  else {  avatar\_img.setImageResource(R.drawable.*boy*);  }  }  @Override  public void afterTextChanged(Editable s) { }  });   //初始化数据库  db=new UsersDBHelper(this);  db.InitialWithTestData();   }   @Override  public void onClick(View v) {  switch (v.getId()){  case R.id.*loginbutton*:  //执行登录操作  login\_action();  break;  case R.id.*register\_tv*:  //跳转到注册界面  Intent intent = new Intent(LoginActivity.this, RegisterActivity.class );  startActivity(intent);  break;  case R.id.*forget*:  //跳转到重设密码界面  Intent intent2 = new Intent(LoginActivity.this, ResetPasswordActivity.class );  startActivity(intent2);  break;  }  }  //登录操作  private void login\_action(){  String name= name\_et.getText().toString();  String password=password\_et.getText().toString();  //先判断数据库里是否存在该用户  if(!db.haveUser(name)){  toast("该用户不存在！");  return;  }  if(db.loginWith(name,password)){  db.close();  // 登录成功，跳转到Home界面  Intent intent = new Intent(LoginActivity.this, HomeActivity.class );  intent.putExtra("name",name);  startActivity(intent);  this.finish();  }  else {  toast("密码错误");  }  }    @Override  protected void onPause() {  super.onPause();  db.close();  }   @Override  protected void onResume() {  super.onResume();  db=new UsersDBHelper(this);  }  //---------toast  public void toast( String text){  Toast toast=Toast.*makeText*(LoginActivity.this, text, Toast.*LENGTH\_SHORT*);  toast.setGravity(Gravity.*CENTER*, 0, 0);  toast.show();  } }  login\_layout.xml  <?xml version="1.0" encoding="utf-8"?> <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:focusableInTouchMode="true"   >   <ImageView  android:id="@+id/pic"  android:layout\_width="370dp"  android:layout\_height="150dp"  android:layout\_centerHorizontal="true"  android:layout\_margin="40dp"  android:scaleType="fitCenter"  android:src="@drawable/boy" />    <EditText  android:id="@+id/Acount"  android:layout\_width="match\_parent"  android:layout\_height="65dp"  android:layout\_below="@+id/pic"  android:layout\_marginLeft="30dp"  android:layout\_marginRight="30dp"  android:layout\_weight="1"   android:background="@drawable/edittext\_selector"  android:hint="@string/account"  android:selectAllOnFocus="false"  android:singleLine="true"  android:textAlignment="center"  android:visibility="visible" />   <EditText  android:id="@+id/passwd"  android:layout\_width="match\_parent"  android:layout\_height="65dp"  android:layout\_below="@+id/Acount"  android:layout\_marginLeft="30dp"  android:layout\_marginTop="10dp"  android:layout\_marginRight="30dp"  android:layout\_marginBottom="35dp"  android:layout\_weight="1"  android:background="@drawable/edittext\_selector"  android:hint="@string/passwd"  android:inputType="textPassword"  android:selectAllOnFocus="false"  android:singleLine="false"  android:textAlignment="center"  android:visibility="visible" />   <Button  android:id="@+id/loginbutton"  android:layout\_width="80dp"  android:layout\_height="50dp"  android:layout\_below="@+id/passwd"  android:layout\_centerHorizontal="true"  android:background="@drawable/button\_selector"  android:text="@string/login" />   <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_alignParentBottom="true"  android:layout\_marginBottom="20dp"   android:orientation="horizontal">   <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_weight="1"  android:orientation="vertical">   <TextView  android:id="@+id/forget"  android:layout\_width="wrap\_content"  android:layout\_height="match\_parent"  android:layout\_gravity="center\_horizontal"  android:hint="@string/forget"  android:textSize="10pt" />   </LinearLayout>   <LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_weight="1"  android:orientation="vertical">   <TextView  android:id="@+id/register\_tv"  android:layout\_width="wrap\_content"  android:layout\_height="match\_parent"  android:layout\_gravity="center\_horizontal"  android:hint="@string/signin"  android:textSize="10pt" />  </LinearLayout>  </LinearLayout>   </RelativeLayout>  RegisterActivity.java  package com.rom471.main;  public class RegisterActivity extends AppCompatActivity implements View.OnClickListener {  Button register\_btn; //注册按钮  EditText name\_et; //用户名输入框  EditText password\_et;//密码输入框  EditText email\_et; //用户邮箱输入框  ImageView avatar\_img; //用户头像图片  UsersDBHelper db; //用户数据库操作类  User User =new User(); //用户对象  public final static int *PICTURE*=10; //通过相册获取用户头像的请求码  public final static int *CAMERA* =11; //通过相机获取用户头像的请求码  boolean userNameValid=true; //记录输入的用户名是否合法  boolean registerFinish=false; //注册是否完成  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_register*);  //绑定组件  register\_btn=findViewById(R.id.*register\_button*);  name\_et=findViewById(R.id.*name\_et*);  password\_et=findViewById(R.id.*passwd\_et*);  email\_et=findViewById(R.id.*email\_et*);  avatar\_img=findViewById(R.id.*avatar\_img*);  //设置点击事件监听器  avatar\_img.setOnClickListener(this);  register\_btn.setOnClickListener(this);  //设置用户名输入框的监视器  name\_et.addTextChangedListener(new TextWatcher() {  @Override  public void beforeTextChanged(CharSequence s, int start, int count, int after) { }  @Override  public void onTextChanged(CharSequence s, int start, int before, int count) {  String name=s.toString();//获取用户输入的用户名  if(db.haveUser(name)){ //该用户名已经存在  name\_et.setTextColor(Color.*RED*);  userNameValid=false;  }  else {  name\_et.setTextColor(Color.*BLACK*);  userNameValid=true;  }  }  @Override  public void afterTextChanged(Editable s) { }  });  //给用户数据库操作类赋值  db=new UsersDBHelper(this);   }   @Override  public void onClick(View v) {  String name=name\_et.getText().toString();  String password=password\_et.getText().toString();  String email=name\_et.getText().toString();  switch (v.getId()){  case R.id.*register\_button*:  if(registerFinish){  Intent intent = new Intent(RegisterActivity.this, HomeActivity.class );  intent.putExtra("name",name);  startActivity(intent);  this.finish();  }  if(!userNameValid){  toast("该用户名已经被使用！");  return;  }  User.setName(name);  User.setPassword(password);  User.setEmail(email);  db.insertUser(User);  MyUtils.*toast*(this,"注册成功");  db.close();  registerFinish=true;  hideViews();  //this.finish();  break;  case R.id.*avatar\_img*:  MyUtils.*verifyStoragePermissions*(this);  getPhotoOrPicture();  break;  }  }  //注册成功后删除一些view  private void hideViews(){  ViewGroup parent =(ViewGroup) name\_et.getParent();  parent.removeView(email\_et);  parent.removeView(password\_et);  name\_et.setFocusable(false);  register\_btn.setText("登录");  }  //弹出对话框，选择从哪里获得头像  private void getPhotoOrPicture(){  final CharSequence[] options = { "从相机", "从相册","取消" };  AlertDialog.Builder builder = new AlertDialog.Builder(RegisterActivity.this);  builder.setTitle("选择一个头像");  builder.setItems(options, new DialogInterface.OnClickListener() {  @Override  public void onClick(DialogInterface dialog, int item) {  if (item==0){ // 打开系统相机获取头像图片  Intent camera = new Intent(MediaStore.*ACTION\_IMAGE\_CAPTURE*);  startActivityForResult(camera, *CAMERA*);  }  else if (item==1){ //打开系统相册获取头像图片  Intent intent = new Intent(Intent.*ACTION\_PICK*,android.provider.MediaStore.Images.Media.*EXTERNAL\_CONTENT\_URI*);  startActivityForResult(intent, *PICTURE*);  }  else if (item==2) { //取消对话框  dialog.dismiss();  }  }  });  builder.show();  }  @Override  protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {  super.onActivityResult(requestCode, resultCode, data);  if(requestCode==*PICTURE* && resultCode==*RESULT\_OK* && data!=null){  String img\_path=null;  Uri uri=data.getData();  img\_path=uri.getPathSegments().get(1);  MyUtils.*log*(img\_path);  Bitmap decodeFile= BitmapFactory.*decodeFile*(img\_path);  Bitmap zoomBitmap = MyUtils.*zoomImg*(decodeFile,200,200);  Drawable avatar\_drawable=new BitmapDrawable(zoomBitmap);  avatar\_img.setImageDrawable(avatar\_drawable);  User.setAvatar(avatar\_drawable);  }  else if(requestCode== *CAMERA* && resultCode==*RESULT\_OK* && data!=null){  Bundle bundle = data.getExtras();  // 获取相机返回的数据，并转换为图片格式  Bitmap bitmap = (Bitmap) bundle.get("data");  Bitmap zoomBitmap = MyUtils.*zoomImg*(bitmap,200,200);  Drawable avatar\_drawable=new BitmapDrawable(zoomBitmap);  avatar\_img.setImageDrawable(avatar\_drawable);  User.setAvatar(avatar\_drawable);  }  }  //---------toast  public void toast( String text){  Toast toast=Toast.*makeText*(RegisterActivity.this, text, Toast.*LENGTH\_SHORT*);  toast.setGravity(Gravity.*CENTER*, 0, 0);  toast.show();  }  }  activity\_register.xml  <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:orientation="vertical"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:focusableInTouchMode="true"  android:gravity="center|top"   >   <ImageView  android:id="@+id/avatar\_img"  android:layout\_height="100dp"  android:layout\_width="100dp"  android:layout\_centerHorizontal="true"  android:layout\_marginVertical="20dp"  android:layout\_marginHorizontal="40dp"  android:src="@drawable/default\_avatar"/>  <EditText  android:id="@+id/name\_et"  android:layout\_width="match\_parent"  android:layout\_height="60dp"  android:layout\_marginVertical="10dp"  android:layout\_marginHorizontal="40dp"  android:background="@drawable/edittext\_selector"  android:hint="@string/name"  android:selectAllOnFocus="false"  android:singleLine="false"  android:visibility="visible"  android:textAlignment="center"/>   <EditText  android:id="@+id/passwd\_et"  android:inputType="textPassword"  android:layout\_width="match\_parent"  android:layout\_height="60dp"  android:layout\_marginVertical="10dp"  android:layout\_marginHorizontal="40dp"  android:background="@drawable/edittext\_selector"  android:hint="@string/passwd"  android:selectAllOnFocus="false"  android:singleLine="false"  android:visibility="visible"  android:textAlignment="center"/>  <EditText  android:id="@+id/email\_et"  android:inputType="textEmailAddress"  android:layout\_width="match\_parent"  android:layout\_height="60dp"  android:layout\_marginVertical="10dp"  android:layout\_marginHorizontal="40dp"  android:background="@drawable/edittext\_selector"  android:hint="@string/email"  android:selectAllOnFocus="false"  android:singleLine="false"  android:visibility="visible"  android:textAlignment="center"/>  <Button  android:id="@+id/register\_button"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginVertical="10dp"  android:layout\_marginHorizontal="40dp"  android:layout\_below="@+id/email\_et"  android:text="@string/register"  android:background="@drawable/button\_selector"/> </LinearLayout>  ResetPasswordActivity.java  package com.rom471.main; import androidx.appcompat.app.AppCompatActivity; import android.os.Bundle; import android.view.View; import android.widget.Button; import android.widget.EditText; import android.widget.ImageView; import com.rom471.lab1.R; import com.rom471.userdb.MyUtils; import com.rom471.userdb.UsersDBHelper;  public class ResetPasswordActivity extends AppCompatActivity implements View.OnClickListener {  Button resetPassword\_btn;  EditText name\_et;  EditText email\_et;  EditText newPassword\_et;  ImageView avatar\_img;  UsersDBHelper db;   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_reset\_password*);  //绑定组件  resetPassword\_btn=findViewById(R.id.*register\_button*);  name\_et=findViewById(R.id.*name\_et*);  newPassword\_et=findViewById(R.id.*passwd\_et*);  email\_et=findViewById(R.id.*email\_et*);  avatar\_img=findViewById(R.id.*avatar\_img*);  //设置监听器  avatar\_img.setOnClickListener(this);  resetPassword\_btn.setOnClickListener(this);  //初始化数据库操作类  db=new UsersDBHelper(this);   }   @Override  public void onClick(View v) {  if (v.getId() == R.id.*register\_button*) {  String name = name\_et.getText().toString();  String password = newPassword\_et.getText().toString();  String email = email\_et.getText().toString();  if(!db.haveUser(name)){  MyUtils.*toast*(this, "该用户不存在！");  return;  }  //修改密码成功会产生变化  int i = db.updatePasswordByNameAndEmail(name, email, password);  if(i<=0){ //没有变化，代表邮箱不匹配  MyUtils.*toast*(this, "邮箱信息不匹配");  return;  }else {  // 密码重置成功  MyUtils.*toast*(this, "密码重设成功");   }    }  }  }  布局文件activity\_reset\_password.xml  <?xml version="1.0" encoding="utf-8"?> <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:focusableInTouchMode="true"  >  <ImageView  android:id="@+id/avatar\_img"  android:layout\_height="120dp"  android:layout\_width="120dp"  android:scaleType="fitXY"  android:layout\_centerHorizontal="true"  android:layout\_margin="40dp"  android:src="@drawable/changepassword"/>  <EditText  android:id="@+id/name\_et"  android:layout\_width="match\_parent"  android:layout\_height="65dp"  android:layout\_marginLeft="30dp"  android:layout\_marginRight="30dp"  android:layout\_below="@+id/avatar\_img"  android:layout\_weight="1"   android:background="@drawable/edittext\_selector"  android:hint="@string/name\_reset"  android:selectAllOnFocus="false"  android:singleLine="false"  android:visibility="visible"  android:textAlignment="center"/>  <EditText  android:id="@+id/email\_et"  android:inputType="textEmailAddress"  android:layout\_width="match\_parent"  android:layout\_height="65dp"  android:layout\_marginTop="10dp"  android:layout\_marginLeft="30dp"  android:layout\_marginRight="30dp"  android:layout\_marginBottom="10dp"  android:layout\_below="@+id/name\_et"  android:layout\_weight="1"  android:background="@drawable/edittext\_selector"  android:hint="@string/email\_reset"  android:selectAllOnFocus="false"  android:singleLine="false"  android:visibility="visible"  android:textAlignment="center"/>  <EditText  android:id="@+id/passwd\_et"  android:inputType="textPassword"  android:layout\_width="match\_parent"  android:layout\_height="65dp"  android:layout\_marginTop="10dp"  android:layout\_marginLeft="30dp"  android:layout\_marginRight="30dp"  android:layout\_marginBottom="10dp"  android:layout\_below="@+id/email\_et"  android:layout\_weight="1"  android:background="@drawable/edittext\_selector"  android:hint="@string/passwd\_new"  android:selectAllOnFocus="false"  android:singleLine="false"  android:visibility="visible"  android:textAlignment="center"/>  <Button  android:id="@+id/register\_button"  android:layout\_width="80dp"  android:layout\_height="50dp"  android:layout\_below="@+id/passwd\_et"  android:layout\_centerHorizontal="true"  android:text="@string/passwd\_reset"  android:background="@drawable/button\_selector"/> </RelativeLayout>  HomeActivity.java  *package* com.rom471.main; *import* android.app.Fragment; *import* android.app.*FragmentManager*; *import* android.app.*FragmentTransaction*; *import* android.content.Intent; *import* android.os.Bundle; *import* android.widget.RadioButton; *import* android.widget.RadioGroup; *import* androidx.appcompat.app.AppCompatActivity; *import* com.rom471.lab1.R; *import* com.rom471.main.fragments.AllAppsFragment; *import* com.rom471.main.fragments.FavoriteFragment; *import* java.util.ArrayList; *import* java.util.*List*;  *public class* HomeActivity *extends* AppCompatActivity *implements* RadioGroup.*OnCheckedChangeListener* {  *public static final* String TAG="cedar";  RadioGroup mRadioGroup;  RadioButton rb1,rb2;   String name; *//登录的用户名  private List*<Fragment> fragments = *new* ArrayList<>();  *private* Fragment fragment;  *private FragmentManager* fm;  *private FragmentTransaction* transaction;   @Override  *protected void* onCreate(Bundle savedInstanceState) {  *super*.onCreate(savedInstanceState);  *//绑定资源* setContentView(R.layout.activity\_home);  mRadioGroup=findViewById(R.id.radioGroup1);  rb1=findViewById(R.id.radio1);  rb2=findViewById(R.id.radio2);  *//初始Fragments* initFragments();  *//设置默认显示fragment* setDefaultFragment();  *//设置监听器* mRadioGroup.setOnCheckedChangeListener(*this*);  }  *//返回登录界面传递的用户名信息  public* String getName(){  Intent intent=getIntent();  name=intent.getStringExtra("name");  *return* name;  }  *//设置默认的fragment为收藏界面  private void* setDefaultFragment(){  fm=getFragmentManager();  transaction=fm.beginTransaction();  fragment=fragments.get(1);  transaction.replace(R.id.mFragment,fragment,"favorite");  transaction.commit();  }  *//初始化Fragments  private void* initFragments(){  fragments.add(*new* AllAppsFragment());  fragments.add(*new* FavoriteFragment());  }   @Override  *public void* onCheckedChanged(RadioGroup group, *int* checkedId) {  transaction=fm.beginTransaction();  *switch* (checkedId){  *case* R.id.radio1:  fragment=fragments.get(0);  transaction.replace(R.id.mFragment,fragment,"all");  *break*;  *case* R.id.radio2:  fragment=fragments.get(1);  transaction.replace(R.id.mFragment,fragment,"favorite");  *break*;  }  transaction.commit();  } }  布局文件layout\_all\_fragment.xml  <?*xml version*="1.0" *encoding*="utf-8"?> <LinearLayout *xmlns:android*="http://schemas.android.com/apk/res/android"  *android:layout\_width*="match\_parent"  *android:layout\_height*="match\_parent"  *android:orientation*="vertical">    <LinearLayout  *android:layout\_width*="wrap\_content"  *android:layout\_height*="wrap\_content"  *android:gravity*="center">   <TextView  *android:id*="@+id/name"  *android:layout\_width*="wrap\_content"  *android:layout\_height*="wrap\_content"  *android:layout\_gravity*="center"   *android:padding*="10dp"  *android:text*="所有应用"  *android:textSize*="20sp" />   <EditText  *android:id*="@+id/search\_box"  *android:layout\_width*="250dp"   *android:layout\_height*="wrap\_content"  *android:layout\_weight*="1"  *android:singleLine*="true"  *android:visibility*="invisible" />   <ImageView  *android:id*="@+id/search\_btn"   *android:layout\_width*="wrap\_content"  *android:layout\_height*="30dp"  *android:layout\_marginRight*="10dp"  *android:layout\_weight*="1"  *android:src*="@drawable/find" />   </LinearLayout>  <TextView  *android:layout\_width*="match\_parent"  *android:layout\_height*="2dp"  *android:background*="#dddddd" />  <ListView  *android:id*="@+id/app\_list"   *android:layout\_width*="match\_parent"  *android:layout\_height*="wrap\_content" />  </LinearLayout>  布局文件layout\_like\_fragment.xml  <?*xml version*="1.0" *encoding*="utf-8"?> <LinearLayout *xmlns:android*="http://schemas.android.com/apk/res/android"  *android:layout\_width*="match\_parent"  *android:layout\_height*="match\_parent"  *android:background*="#F7F7F7"  *android:orientation*="vertical">   <TextView  *android:id*="@+id/like\_tile"  *android:layout\_width*="match\_parent"  *android:layout\_height*="wrap\_content"  *android:layout\_marginVertical*="10dp"  *android:gravity*="center"  *android:textSize*="20dp" />   <TextView  *android:layout\_width*="match\_parent"  *android:layout\_height*="2dp"  *android:background*="#dddddd" />   <androidx.recyclerview.widget.RecyclerView  *android:id*="@+id/users\_listview"  *android:layout\_width*="match\_parent"  *android:layout\_height*="match\_parent" />   <LinearLayout  *android:id*="@+id/empty\_view"  *android:layout\_width*="match\_parent"  *android:layout\_height*="wrap\_content"  *android:layout\_marginVertical*="40dp"  *android:gravity*="center"  *android:orientation*="vertical">   <ImageView  *android:id*="@+id/radio1"  *android:layout\_width*="100dp"  *android:layout\_height*="wrap\_content"  *android:src*="@drawable/apps\_selector" />   <TextView  *android:layout\_width*="match\_parent"  *android:layout\_height*="wrap\_content"   *android:gravity*="center"  *android:text*="@string/empty\_hint"  *android:textSize*="20dp" />   </LinearLayout>  </LinearLayout>  FavoriteFragment.java  package com.rom471.main.fragments;  import android.app.Fragment; import android.os.Bundle; import android.view.LayoutInflater; import android.view.View; import android.view.ViewGroup; import android.widget.TextView; import androidx.annotation.NonNull; import androidx.annotation.Nullable; import androidx.recyclerview.widget.DefaultItemAnimator; import androidx.recyclerview.widget.GridLayoutManager; import androidx.recyclerview.widget.RecyclerView; import com.rom471.adapters.FavoriteListAdapter; import com.rom471.appsdb.AppBean; import com.rom471.lab1.R; import com.rom471.main.HomeActivity;  public class FavoriteFragment extends Fragment {  RecyclerView favoriteListview; //列表视图  TextView tile\_tv;  FavoriteListAdapter mAdapter;  public FavoriteFragment(){  }  @Nullable  @Override  public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {  return inflater.inflate(R.layout.*layout\_like\_fragment*, container, false);  }  @Override  public void onActivityCreated(@Nullable Bundle savedInstanceState) {  super.onActivityCreated(savedInstanceState);  //绑定资源  tile\_tv=getActivity().findViewById(R.id.*like\_tile*);  String userName = ((HomeActivity) getActivity()).getName();  String db\_name=userName+".db";  favoriteListview =getActivity().findViewById(R.id.*users\_listview*);  mAdapter=new FavoriteListAdapter(getContext(),db\_name);  favoriteListview.setLayoutManager(new GridLayoutManager(getContext(),2));  favoriteListview.setAdapter(mAdapter);  //给recyclerview设置动画  DefaultItemAnimator defaultItemAnimator = new DefaultItemAnimator();  defaultItemAnimator.setRemoveDuration(300);  favoriteListview.setItemAnimator(defaultItemAnimator);  //设置标题标签  tile\_tv.setText(userName+"的收藏列表");  }  public boolean hasAppName(String appname){  return mAdapter.hasAppName(appname);  }  public void insertApp(AppBean app){  mAdapter.insertApp(app);  } }  layout\_favorite\_fragment.xml  <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:background="#F7F7F7"  android:orientation="vertical">   <TextView  android:id="@+id/like\_tile"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginVertical="10dp"  android:gravity="center"  android:textSize="20dp" />   <TextView  android:layout\_width="match\_parent"  android:layout\_height="2dp"  android:background="#dddddd" />   <androidx.recyclerview.widget.RecyclerView  android:id="@+id/users\_listview"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent" />   <LinearLayout  android:id="@+id/empty\_view"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginVertical="40dp"  android:gravity="center"  android:orientation="vertical">   <ImageView  android:id="@+id/radio1"  android:layout\_width="100dp"  android:layout\_height="wrap\_content"  android:src="@drawable/apps\_selector" />   <TextView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"   android:gravity="center"  android:text="@string/empty\_hint"  android:textSize="20dp" />   </LinearLayout>  </LinearLayout>  AllAppsFragment.java  public class AllAppsFragment extends Fragment implements View.OnClickListener, TextWatcher {  AllListAdapter appAdapter;  ListView app\_list\_view;  ImageView search\_btn;  EditText search\_box;  PackageManager pm;  FavoriteFragment favoriteFragment;  @Nullable  @Override  public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {  return inflater.inflate(R.layout.*layout\_all\_fragment*,container,false);   }  @Override  public void onActivityCreated(@Nullable Bundle savedInstanceState) {  super.onActivityCreated(savedInstanceState);  FragmentManager fragmentManager=getFragmentManager();  favoriteFragment=(FavoriteFragment) fragmentManager.findFragmentByTag("favorite");  search\_btn=getActivity().findViewById(R.id.*search\_btn*);  search\_box=getActivity().findViewById(R.id.*search\_box*);  search\_btn.setOnClickListener(this);  search\_box.addTextChangedListener(this);  app\_list\_view=getActivity().findViewById(R.id.*app\_list*);  appAdapter =new AllListAdapter(getContext());  pm=getContext().getPackageManager();  getApplist();  app\_list\_view.setOnItemClickListener(new AdapterView.OnItemClickListener() {  @Override  public void onItemClick(AdapterView<?> parent, View view, int position, long id) {  AppBean app=appAdapter.get(position);  if(favoriteFragment!=null && favoriteFragment.hasAppName(app.getAppname())){  Toast.*makeText*(getContext(),"该应用已经被收藏",Toast.*LENGTH\_SHORT*).show();  return;  }  try {  favoriteFragment.insertApp(app);  Animation animation = AnimationUtils.*loadAnimation*(getContext(),R.anim.*item\_add\_anim*);  view.startAnimation(animation);  appAdapter.notifyDataSetChanged();  }catch (Exception e){  e.printStackTrace();   }  }  });  }  private void getApplist(){   List<PackageInfo> packages=pm.getInstalledPackages(0);  appAdapter =new AllListAdapter(getContext());  for(PackageInfo packageInfo:packages){  if((packageInfo.applicationInfo.flags& ApplicationInfo.*FLAG\_SYSTEM*)==0) {//非系统应用  ApplicationInfo info = packageInfo.applicationInfo;  AppBean app=new AppBean();  String name = info.loadLabel(pm).toString();  app.setAppname(name);  app.setPkgname(info.packageName);  app.setIcon(info.loadIcon(pm));  app.setLiked(favoriteFragment.hasAppName(name));  appAdapter.add(app);  }  else{ //系统应用，暂时不添加 // ApplicationInfo info = new ApplicationInfo(); // info.name = packageInfo.applicationInfo.loadLabel(pm).toString(); // AppBean app=new AppBean(); // String name = info.loadLabel(pm).toString(); // app.setAppname(name); // app.setPkgname(info.packageName); // app.setIcon(info.loadIcon(pm)); // app.setLiked(favoriteFragment.hasAppName(name)); // appAdapter.add(app);  }  }  app\_list\_view.setAdapter(appAdapter);  }     @Override  public void onClick(View v) {  switch (v.getId()){  case R.id.*search\_btn*:  search\_box.setVisibility(View.*VISIBLE*);  break;  }  }  @Override  public void beforeTextChanged(CharSequence s, int start, int count, int after) {   }  @Override  public void onTextChanged(CharSequence s, int start, int before, int count) {  appAdapter.getFilter().filter(s);  }   @Override  public void afterTextChanged(Editable s) {   }  }  layout\_all\_fragment.xml  <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical">  <LinearLayout  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:gravity="center">  <TextView  android:id="@+id/name"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="center"  android:padding="10dp"  android:text="所有应用"  android:textSize="20sp" />   <EditText  android:id="@+id/search\_box"  android:layout\_width="250dp"  android:layout\_height="wrap\_content"  android:layout\_weight="1"  android:singleLine="true"  android:visibility="invisible" />   <ImageView  android:id="@+id/search\_btn"  android:layout\_width="wrap\_content"  android:layout\_height="30dp"  android:layout\_marginRight="10dp"  android:layout\_weight="1"  android:src="@drawable/find" />  </LinearLayout>  <TextView  android:layout\_width="match\_parent"  android:layout\_height="2dp"  android:background="#dddddd" />  <ListView  android:id="@+id/app\_list"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content" /> </LinearLayout>  FavoriteListAdapter.java  public class FavoriteListAdapter extends RecyclerView.Adapter<RecyclerView.ViewHolder> {  private List<AppBean> mAppsList;  private final int SHOW\_MENU = 1;  private final int HIDE\_MENU = 2;  AppDAO dao; //用户收藏数据库操作类  Context context;  PackageManager pm;  public FavoriteListAdapter(Context context, String db\_name) {  this.context=context;  this.mAppsList = mAppsList;  dao=getDao(db\_name);  initApps();  }  @NonNull  @Override  public RecyclerView.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  View view;  if(viewType==SHOW\_MENU){  view= LayoutInflater.*from*(parent.getContext()).inflate(R.layout.*favorite\_app\_item\_menu*,parent,false);  return new MenuViewHolder((view));  }  else {  view= LayoutInflater.*from*(parent.getContext()).inflate(R.layout.*favorite\_app\_item*,parent,false);  ViewHolder holder=new ViewHolder(view);  return holder;  }  }  public void showMenu(int position) {  for(int i=0; i<mAppsList.size(); i++){  mAppsList.get(i).setShowMenu(false);  }  mAppsList.get(position).setShowMenu(true);  notifyDataSetChanged();  }  public void closeMenu() {  for(int i=0; i<mAppsList.size(); i++){  mAppsList.get(i).setShowMenu(false);  }  notifyDataSetChanged();  }  //渲染数据到View中  @Override  public void onBindViewHolder(RecyclerView.ViewHolder holder, final int position) {  //默认项目  if(holder instanceof ViewHolder){  ViewHolder mHolder=(ViewHolder) holder;  AppBean app=mAppsList.get(position);  mHolder.appIcon.setBackground(app.getIcon());  mHolder.appName.setText(app.getAppname());  mHolder.list.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  if(position<mAppsList.size())  showMenu(position);   }  });  }  //项目菜单  if(holder instanceof MenuViewHolder){  MenuViewHolder mHolder=(MenuViewHolder) holder;  AppBean app=mAppsList.get(position);  mHolder.appName.setText(app.getAppname());  mHolder.list.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  closeMenu();  }  });  mHolder.delete.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  dao.deleteApp(mAppsList.get(position));  mAppsList.remove(position);  notifyItemRemoved(position);  notifyItemRangeChanged(position,mAppsList.size()-position);  }  });  mHolder.open.setOnClickListener(new MyOnclickListenr(context,mAppsList.get(position).getPkgname()));  }  }  //可以接受参数的点击监听器  private class MyOnclickListenr implements View.OnClickListener{  Context context;  String pkgname;  public MyOnclickListenr(Context context,String pkgname){  this.context=context;  this.pkgname=pkgname;  }  @Override  public void onClick(View v) {  try {  Intent intent =context.getPackageManager().getLaunchIntentForPackage(pkgname);  context.startActivity(intent);  toast("启动成功");  }catch (Exception e){  toast("打开失败");  }  }  }  @Override  public int getItemViewType(int position) {  if(mAppsList.get(position).isShowMenu()){  return SHOW\_MENU;  }else{  return HIDE\_MENU;  }  }  @Override  public int getItemCount() {  return mAppsList.size();  }  static class ViewHolder extends RecyclerView.ViewHolder {  ImageView appIcon;  TextView appName;  LinearLayout list;  public ViewHolder(View view) {  super(view);  appIcon = (ImageView) view.findViewById(R.id.*item\_icon*);  appName = (TextView) view.findViewById(R.id.*item\_name*);  list=view.findViewById(R.id.*item\_list\_layout*);   }  }  public class MenuViewHolder extends RecyclerView.ViewHolder{  TextView appName;  LinearLayout list;  Button delete;  Button open;   public MenuViewHolder(View view){  super(view);  list=view.findViewById(R.id.*item\_list\_layout*);  appName = (TextView) view.findViewById(R.id.*item\_name*);  delete=(Button) view.findViewById(R.id.*recycle\_menu\_delete*);  open=(Button) view.findViewById(R.id.*recycle\_menu\_open*);   }  }  //获得收藏应用数据库操作对象  private AppDAO getDao(String db\_name){  AppDataBase db = Room.*databaseBuilder*(context, AppDataBase.class, db\_name).allowMainThreadQueries().build();  return db.getAppDAO();  }  private void initApps(){  pm=context.getPackageManager();  mAppsList=dao.getAllApps();  for (AppBean app:mAppsList  ) {  try {  ApplicationInfo appInfo = pm.getApplicationInfo(app.getPkgname(), PackageManager.*GET\_META\_DATA*);   Drawable appIcon = pm.getApplicationIcon(appInfo);  app.setIcon(appIcon);  app.setShowMenu(false);   } catch (PackageManager.NameNotFoundException e) {  e.printStackTrace();  }  }  }  public boolean hasAppName(String appname){   for (AppBean app:  mAppsList) {  if(app.getAppname().equals(appname)){   return true;  }  }  return false;  }  //添加进收藏  public void insertApp(AppBean app){  app.setLiked(true);  mAppsList.add(app);  dao.insertApp(app);  }  public void toast( String text){  Toast toast=Toast.*makeText*(context, text, Toast.*LENGTH\_SHORT*);  toast.setGravity(Gravity.*CENTER*, 0, 0);  toast.show();  } }  AllListAdapter.java  public class AllListAdapter extends BaseAdapter implements Filterable {  public static final String *TAG*= AllListAdapter.class.getSimpleName();  private List<AppBean> appList;  private List<AppBean> allAppList;  private LayoutInflater mInflater;  PackageManager pm;  public AllListAdapter(Context context){  appList =new ArrayList<>();  mInflater=LayoutInflater.*from*(context);  pm=context.getPackageManager();  }    public void add(AppBean app){  this.appList.add(app);  }  @Override  public int getCount() {  return appList ==null?0: appList.size();  }  @Override  public Object getItem(int position) {  return appList ==null?null: appList.get(position);  }  @Override  public long getItemId(int position) {  return position;  }  @RequiresApi(api = Build.VERSION\_CODES.*Q*)  @Override  public View getView(int position, View convertView, ViewGroup parent) {  ViewHolder holder=null;  if(convertView==null){  holder=new ViewHolder();  convertView=mInflater.inflate(R.layout.*app\_list\_layout*,null);  bindView(convertView,holder);  convertView.setTag(holder);  }else {  holder= (ViewHolder) convertView.getTag();  }  renderView(position,holder);  return convertView;   }  public void bindView(View convertView,ViewHolder holder){  holder.icon=convertView.findViewById(R.id.*appicon*);  holder.liked =convertView.findViewById(R.id.*liked*);  holder.appname=convertView.findViewById(R.id.*appname*);  }  @RequiresApi(api = Build.VERSION\_CODES.*Q*)  public void renderView(int position, ViewHolder holder){  AppBean appinfo= appList.get(position);  if(appinfo.isLiked()){  holder.liked.setBackgroundResource(R.drawable.*stared*);  }else {  holder.liked.setBackgroundResource(R.drawable.*star*);  }  Drawable icon=appinfo.getIcon();  holder.icon.setBackground(icon);  holder.appname.setText(appinfo.getAppname());   }   public AppBean get(int position) {  return appList.get(position);  }   @Override  public Filter getFilter() {  if(allAppList==null){  allAppList=appList;  }  Filter filter=new Filter() {  @Override  protected FilterResults performFiltering(CharSequence constraint) {  FilterResults results = new FilterResults();  ArrayList<AppBean> filteredApps = new ArrayList<>();    constraint = constraint.toString().toLowerCase();  for (int i = 0; i < allAppList.size(); i++) {  AppBean app = allAppList.get(i);  if (app.getAppname().toLowerCase().contains(constraint.toString())) {  filteredApps.add(app);  }  }  results.count =filteredApps.size();  results.values = filteredApps;  return results;  }   @Override  protected void publishResults(CharSequence constraint, FilterResults results) {  appList = (List<AppBean>) results.values;  notifyDataSetChanged();  }  };  return filter;  }  class ViewHolder{  public ImageView icon; //显示应用图标  public ImageView liked;//显示受否被收藏  public TextView appname;//显示应用标签  } }  与用户数据库操作有关的类  User.java  package com.rom471.userdb; import android.content.Context; import android.content.res.Resources; import android.graphics.Bitmap; import android.graphics.BitmapFactory; import android.graphics.drawable.BitmapDrawable; import android.graphics.drawable.Drawable; import java.io.ByteArrayOutputStream; public class User {  private int id;  private String name;  private String password;  private String email;  private Drawable avatar;  public void setAvatar(Drawable avatar) {  this.avatar = avatar;  }  public User() {  this.name = null;  this.password = null;  this.email = null;  }  @Override  public String toString() {  return "User{" +  "id=" + id +  ", name='" + name + '\'' +  ", password='" + password + '\'' +  ", avatarUrl='" + email + '\'' +  '}';  }   public User( String name, String password, String email) {   this.name = name;  this.password = password;  this.email = email;  }  //指定资源id，来设置用户的头像  public void setAvatarResource(Context context, int id){  Resources resources=context.getResources();  Bitmap bmp= BitmapFactory.*decodeResource*(resources, id);  ByteArrayOutputStream baos=new ByteArrayOutputStream();  bmp.compress(Bitmap.CompressFormat.*PNG*,100,baos);//压缩图片  byte[] bytes = baos.toByteArray();  this.avatar = *byteToDrawable*(bytes);  }  public Drawable getAvatar() {  return avatar;  }  public byte[] getAvatarBytes(){  return MyUtils.*drawableToByteArray*(this.avatar);  }  public static Drawable byteToDrawable(byte[] data) {  if (data == null)  return null;  else  return new BitmapDrawable(BitmapFactory.*decodeByteArray*(data, 0, data.length));  }  public int getId() {  return id;  }   public void setId(int id) {  this.id = id;  }   public String getName() {  return name;  }   public String getPassword() {  return password;  }   public void setPassword(String password) {  this.password = password;  }   public void setName(String name) {  this.name = name;  }   public String getEmail() {  return email;  }   public void setEmail(String email) {  this.email = email;  } }  UsersDBHelper.java  package com.rom471.userdb;  import android.content.ContentValues; import android.content.Context; import android.database.Cursor; import android.database.sqlite.SQLiteDatabase; import android.database.sqlite.SQLiteOpenHelper; import android.graphics.BitmapFactory; import android.graphics.drawable.BitmapDrawable; import android.graphics.drawable.Drawable; import com.rom471.lab1.R; import java.util.ArrayList; import java.util.List;  public class UsersDBHelper extends SQLiteOpenHelper {  public static final String *TAG*="cedar";  //数据库名字  public static final String *DB\_NAME*="users.db";  //存放用户信息的表名  public static final String *TABLE\_NAME*="User";  public static final int *DB\_VERSION*=1;  //内部Context引用  private Context context;  //内部数据库引用  private SQLiteDatabase db;  public void close(){  db.close();  super.close();  }  @Override  public void onCreate(SQLiteDatabase db) {  }  @Override  public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  }  //初始化内部引用  public UsersDBHelper(Context context){  super(context,*DB\_NAME*,null,*DB\_VERSION*);  db=super.getWritableDatabase();  this.context=context;  }  //创建用户信息表  public void createUserTable(){  final String CREATE\_USER="create table User ("  + "uid integer primary key autoincrement,"  + "name text ,"  + "password text,"  + "email text,"  + "avatar blob)";  db.execSQL(CREATE\_USER);  }   public void dropTable(String table\_name){  db.execSQL("drop table if exists "+table\_name);  }  //判断是否存在表  public boolean havingTable(String table\_name){  Cursor cursor;  boolean a=false;  cursor = db.rawQuery("select name from sqlite\_master where type='table' ", null);  while(cursor.moveToNext()) {  //遍历出表名  String name = cursor.getString(0);  if (name.equals(table\_name)) { //存在表  cursor = db.query(table\_name, null, null, null, null, null, null);  //检查是不是空表  if (cursor.getCount() > 0) {  cursor.close();  return true;  } else //空表  {  cursor.close();  return false;  }  }  }  cursor.close();  return false;  }   //用测试数据初始化数据库  public void InitialWithTestData(){  if(havingTable(*TABLE\_NAME*))  return;  createUserTable();  addUserWithAvatar("xxs","cedar","527@qq.com",R.drawable.*avatar\_xxs*);  addUserWithAvatar("wjf","wjf","wjf@qq.com",R.drawable.*avatar\_wjf*);   addUserWithAvatar("wkq","wkq",null,R.drawable.*avatar3*);   }  //添加带头像的用户  private void addUserWithAvatar(String name,String password,String email,int rid){  User User =new User(name,password,email);  User.setAvatarResource(context,rid);  this.insertUser(User);  }  private void addUserBasic(String name,String password){  User User =new User(name,password,null);  this.insertUser(User);  }  //向表中插入一个用户  public void insertUser(User User){  ContentValues contentValues=new ContentValues();  //contentValues.put("id",user.getId());  contentValues.put("name", User.getName());  contentValues.put("password", User.getPassword());  contentValues.put("email", User.getEmail());  contentValues.put("avatar", User.getAvatarBytes());  db.insert(*TABLE\_NAME*,null,contentValues);  }  //查询所有用户并返回  public List<User> queryAll(){  List<User> Users =new ArrayList<>();  String sql = "SELECT uid,name,password,email,avatar FROM "+*TABLE\_NAME* +" ";  Cursor cursor=db.rawQuery(sql,null);   while(cursor.moveToNext()){  int id=cursor.getInt(cursor.getColumnIndex("uid"));  String name=cursor.getString(cursor.getColumnIndex("name"));  String password=cursor.getString(cursor.getColumnIndex("password"));  String email=cursor.getString(cursor.getColumnIndex("email"));  User User =new User(name,password,email);  User.setId(id);  byte [] bytes=null;  Drawable avatar=null;  bytes=cursor.getBlob(cursor.getColumnIndex("avatar"));  if(bytes!=null){  avatar= new BitmapDrawable(BitmapFactory.*decodeByteArray*(bytes, 0, bytes.length));  User.setAvatar(avatar);  }  Users.add(User);  }  cursor.close();  return Users;  }  //判断用户名是否存在  public boolean haveUser(String name){  String sql=" select \* from "+*TABLE\_NAME*+ " where name = ?";  Cursor cursor=db.rawQuery(sql,new String[]{name});  if(cursor.moveToFirst()==true){  cursor.close();  return true;  }  cursor.close();  return false;  }  //使用用户名和密码登录  public boolean loginWith(String name,String password){  String sql=" select \* from "+*TABLE\_NAME*+ " where name = ? and password=?";  Cursor cursor=db.rawQuery(sql,new String[]{name,password});  if(cursor.moveToFirst()==true){  cursor.close();  return true;  }  cursor.close();  return false;  }  //由用户名获取头像  public Drawable getAvatarByName(String name){  String sql = "SELECT avatar FROM "+*TABLE\_NAME* +" WHERE name = ?";  Drawable avatar=null;  byte[] bytes;  Cursor cursor=db.rawQuery(sql,new String[]{name});  if(cursor.moveToFirst()){  if((bytes=cursor.getBlob(cursor.getColumnIndex("avatar")))!=null){  avatar= new BitmapDrawable(BitmapFactory.*decodeByteArray*(bytes, 0, bytes.length));  }  }  cursor.close();  return avatar;  }  //提供用户名，修改密码,返回数据库中有几行被修改了。  public int updatePasswordByName(String name,String password){  ContentValues values=new ContentValues();  values.put("password",password);  int update = db.update(*TABLE\_NAME*, values, "name = ?", new String[]{name});  return update;  }  //提供用户名和邮箱，修改密码,返回数据库中有几行被修改了。  public int updatePasswordByNameAndEmail(String name,String email,String password){  ContentValues values=new ContentValues();  values.put("password",password);  int update = db.update(*TABLE\_NAME*, values, "name = ? and email = ?", new String[]{name ,email});  return update;  }  }  与应用收藏有关的数据库操作类  AppBean.java  package com.rom471.appsdb;  import android.graphics.drawable.Drawable; import androidx.room.Entity; import androidx.room.Ignore; import androidx.room.Index; import androidx.room.PrimaryKey; @Entity(indices = {  @Index(value = "id"),  @Index(value = "pkgname",unique = true) }) public class AppBean {  @PrimaryKey(autoGenerate = true)  private int id;  String appname;  String pkgname;  @Ignore  Drawable icon;  @Ignore  private boolean showMenu = false;  @Ignore  private boolean liked=false;   public boolean isLiked() {  return liked;  }   public void setLiked(boolean liked) {  this.liked = liked;  }   public boolean isShowMenu() {  return showMenu;  }   public void setShowMenu(boolean showMenu) {  this.showMenu = showMenu;  }   public Drawable getIcon() {  return icon;  }   public void setIcon(Drawable icon) {  this.icon = icon;  }   public AppBean() {  }   public String getAppname() {  return appname;  }   public int getId() {  return id;  }   public void setId(int id) {  this.id = id;  }   public void setAppname(String appname) {  this.appname = appname;  }   public String getPkgname() {  return pkgname;  }   public void setPkgname(String pkgname) {  this.pkgname = pkgname;  } }  AppDAO.java  package com.rom471.appsdb;  import androidx.room.Dao; import androidx.room.Delete; import androidx.room.Insert; import androidx.room.Query; import androidx.room.Update; import java.util.List; @Dao public interface AppDAO {  @Insert  void insertApp(AppBean...appBeans);   @Update  void updateApp(AppBean...appBeans);  @Delete  void deleteApp(AppBean...appBeans);   @Query("delete from AppBean")  void deleteALl();  @Query("select \* from AppBean order by id desc")  List<AppBean> getAllApps(); }  AppDataBase.java  package com.rom471.appsdb;  import androidx.room.Database; import androidx.room.RoomDatabase; @Database(entities = {AppBean.class},version = 1,exportSchema = false) public abstract class AppDataBase extends RoomDatabase {  public abstract AppDAO getAppDAO(); }  工具类MyUtils.java  package com.rom471.userdb;  import android.app.Activity; import android.content.Context; import android.content.pm.PackageManager; import android.graphics.Bitmap; import android.graphics.Matrix; import android.graphics.drawable.BitmapDrawable; import android.graphics.drawable.Drawable; import android.util.Log; import android.view.Gravity; import android.widget.Toast;  import androidx.core.app.ActivityCompat;  import java.io.ByteArrayOutputStream;  public class MyUtils {  public static Bitmap zoomImg(Bitmap bm, int newWidth , int newHeight){  // 获得图片的宽高  int width = bm.getWidth();  int height = bm.getHeight();  // 计算缩放比例  float scaleWidth = ((float) newWidth) / width;  float scaleHeight = ((float) newHeight) / height;  // 取得想要缩放的matrix参数  Matrix matrix = new Matrix();  matrix.postScale(scaleWidth, scaleHeight);  // 得到新的图片  Bitmap newbm = Bitmap.*createBitmap*(bm, 0, 0, width, height, matrix, true);  return newbm;  }   public static void toast(Context context, String text){  Toast toast=Toast.*makeText*(context, text, Toast.*LENGTH\_SHORT*);  toast.setGravity(Gravity.*CENTER*, 0, 0);  toast.show();  }   public static byte[] drawableToByteArray(Drawable d) {   if (d != null) {  Bitmap imageBitmap = ((BitmapDrawable) d).getBitmap();  ByteArrayOutputStream baos = new ByteArrayOutputStream();  imageBitmap.compress(Bitmap.CompressFormat.*PNG*, 100, baos);  byte[] byteData = baos.toByteArray();   return byteData;  } else  return null;   }  public static void log(String...strs){  StringBuilder sb=new StringBuilder();  for (String str:strs  ) {  sb.append(str).append(" ");  }  Log.*d*("cedar", sb.toString());  }   //先定义  private static final int *REQUEST\_EXTERNAL\_STORAGE* = 1;   private static String[] *PERMISSIONS\_STORAGE* = {  "android.permission.READ\_EXTERNAL\_STORAGE",  "android.permission.WRITE\_EXTERNAL\_STORAGE" };   //然后通过一个函数来申请  public static void verifyStoragePermissions(Activity activity) {  try {  //检测是否有写的权限  int permission = ActivityCompat.*checkSelfPermission*(activity,  "android.permission.WRITE\_EXTERNAL\_STORAGE");  if (permission != PackageManager.*PERMISSION\_GRANTED*) {  // 没有写的权限，去申请写的权限，会弹出对话框  ActivityCompat.*requestPermissions*(activity, *PERMISSIONS\_STORAGE*,*REQUEST\_EXTERNAL\_STORAGE*);  }  } catch (Exception e) {  e.printStackTrace();  }  } } |