E-Learning System 设计文档

Assignment 3

小组成员	
17301088	包天活
17301101	刘彬

一、第三次作业前端界面在第二次的基础上做了优化

1、 首先定义类两个 viewitemlayouts 来展示课程列表，课程列表 recyclerview 增加 viewHolder 使用不同的 viewitem layouts 展示不同课程，展示课程的类型根据才课程包含材料的种类，界面实现如下

recyclerview 对应适配器里的关键代码如下

```java
public static class ViewHolder extends RecyclerView.ViewHolder {
    public ImageView imageView;
    public TextView tvTitle,tvTime,tvContext;

    public ViewHolder(View v) {
        super(v);
        imageView = (ImageView)v.findViewById(R.id.IV_list_Id);
        tvTitle = (TextView)v.findViewById(R.id.TV_listTitle_Id);
        tvTime = (TextView)v.findViewById(R.id.TV_listTime_Id);
        tvContext = (TextView)v.findViewById(R.id.TV_listContext_Id);
    }
}

public static class MoreViewHolder extends RecyclerView.ViewHolder {
    public ImageView imageVideo;
    public ImageView imageBook;
    public TextView tvCourse,tvSchool,tvTeacher;

    public MoreViewHolder(View v) {
        super(v);
        imageVideo = (ImageView)v.findViewById(R.id.IV_video);
        imageBook = (ImageView)v.findViewById(R.id.IV_book);
        tvCourse = (TextView)v.findViewById(R.id.TV_course);
        tvSchool = (TextView)v.findViewById(R.id.TV_school);
        tvTeacher = (TextView)v.findViewById(R.id.TV_teacher);
    }
}
```

```java
public void onBindViewHolder(RecyclerView.ViewHolder holder,final int position) {
    Log.d(TAG,  msg: "onBindViewHolder: ");
    course = mCourseList.get(position);

    if (holder instanceof MoreViewHolder) {
        ((MoreViewHolder)holder).tvCourse.setText(course.getcourseName());
        ((MoreViewHolder)holder).tvTeacher.setText(course.getTeacher());
        ((MoreViewHolder)holder).tvSchool.setText(course.getSchool());
        Picasso.get()
                .load(course.getImageURL())
                .into(((MoreViewHolder)holder).imageBook);

    }else if(holder instanceof ViewHolder){
        ((ViewHolder)holder).tvTitle.setText(course.getcourseName());
        ((ViewHolder)holder).tvTime.setText(course.getTeacher());
        ((ViewHolder)holder).tvContext.setText(course.getSchool());
        Picasso.get()
                .load(course.getImageURL())
                .into(((ViewHolder)holder).imageView);
    }

    holder.itemView.setOnClickListener((v) -> {
            Intent intent = new Intent(context, CourseDetailsActivity.class);
            String idContent = mCourseList.get(position).getCourseID();
            intent.putExtra( name: "idContent",idContent);
            context.startActivity(intent);
    });
}
```

2、 登录 APP 后，保存登录状态，如果不退出登录，则下次打开 APP 进入课程列表界面，如果退出登录，则跳转到登录界面。

登录状态保存使用到了 SharedPreferences，我在代码里封装了一个 SharedPreferencesUtil 类，使用它来保存登录状态。

```
public class SharedPreferencesUtil {
    private static final String TAG="TAG";
    private static final String KEY_LOGIN="KEY_LOGIN";

    private static SharedPreferences mPreference;
    private static SharedPreferences.Editor mEditor;
    private static SharedPreferencesUtil mSharedPreferenceUtil;
    private final Context context;

    public SharedPreferencesUtil(Context context){
        this.context=context.getApplicationContext();
        mPreference=this.context.getSharedPreferences(TAG,Context.MODE_PRIVATE);
        mEditor=mPreference.edit();
    }

    //简单的单例实现
    public static SharedPreferencesUtil getInstance(Context context){
        if(mSharedPreferenceUtil==null){
            mSharedPreferenceUtil = new SharedPreferencesUtil(context);
        }
        return mSharedPreferenceUtil;
    }

    public boolean isLogin() { return getBoolean(KEY_LOGIN, defaultValue: false); }

    public void setLogin(boolean value) { putBoolean(KEY_LOGIN,value); }

    private void put(String key,String value){
        mEditor.putString(key,value);
        mEditor.commit();
    }
```

二、android 数据库

3、 SQLite 实现用户登录信息持久化

1) 功能描述：

用户在登录时输入邮箱和密码，点击登录按钮后，程序将邮箱和密码存入到 SQLite 数据库中。当用户退出登录后再重新登录时，账号和密码直接预填写到相应输入框。此功能方便用户快速再次登录。

2) 封装 MyDatabaseHelper 类

添加构造器，并重写实现两个抽象方法。

```java
public class MyDatabaseHelper extends SQLiteOpenHelper {
    public static final String DATABASE_NAME = "eLearning.db";
    private static final String TAG = "MyDatabaseHelper";

    private static final String CREATE_TABLE_user = "CREATE TABLE user(\n" +
                                                    "    mail text PRIMARY KEY,\n" +
                                                    "    password text\n" +
                                                    ")";

    private Context mContext;

    public MyDatabaseHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version){
        super(context, name, factory, version);
        mContext = context;
    }

    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {
        sqLiteDatabase.execSQL(CREATE_TABLE_user);
        Log.d(TAG,  msg: "onCreate: table user");
    }

    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int oldVersion, int newVersion) {
        sqLiteDatabase.execSQL("DROP TABLE IF EXISTS user");
        onCreate(sqLiteDatabase);
        Log.d(TAG,  msg: "onUpgrade: ");
    }
}
```

3) 在登录时判断数据库中是否存在邮箱和密码，若存在则预填到输入框内。

```java
//实例化MyDataHelper
databaseHelper = new MyDatabaseHelper( context: this, MyDatabaseHelper.DATABASE_NAME,  factory: null,  version: 1);
//将已有的用户邮箱和密码展示到页面上(实现记住账号和密码功能)
SQLiteDatabase database = databaseHelper.getWritableDatabase();
Cursor cursor = database.rawQuery( sql: "SELECT * FROM user",  selectionArgs: null);
if(cursor.moveToFirst()){
    String mail = cursor.getString(cursor.getColumnIndex( s: "mail"));
    String password = cursor.getString(cursor.getColumnIndex( s: "password"));

    et_username.setText(mail);
    et_password.setText(password);
}
```

4) 用户点击登录后，程序为用户持久化记录邮箱和密码。

```
SQLiteDatabase database = databaseHelper.getWritableDatabase();
Cursor cursor = database.rawQuery( sql: "SELECT mail FROM user",  selectionArgs: null);
boolean exist = false;
if(cursor.moveToFirst()){
    do{
        if(user.equals(cursor.getString(cursor.getColumnIndex( s: "mail")))){
            exist = true;
            break;
        }
    }while (cursor.moveToNext());
}

if(!exist){
    database.execSQL( sql: "INSERT INTO user (mail, password) VALUES (?, ?)",
            new String[] {user, pass});
}
```

4、 LitePal 实现课程信息本地持久化

1) 功能描述：

用户登录成功后，进入课程列表页面，程序会向服务器请求课程列表资源，请求成功后，程序通过 LitePal 将课程数据存入到本地数据库中。当用户进入其他页面再回到该页面时，直接进行展示而不是再次从网络中再次请求。

2) LitePal 简介：

LitePal 是一个 Android 开源库，它使开发者使用 SQLite 数据库变得非常容易。开发者可以不用写一句 SQL 语句就可以完成大部分数据库操作，包括创建表，更新表，约束操作，聚合功能等等。

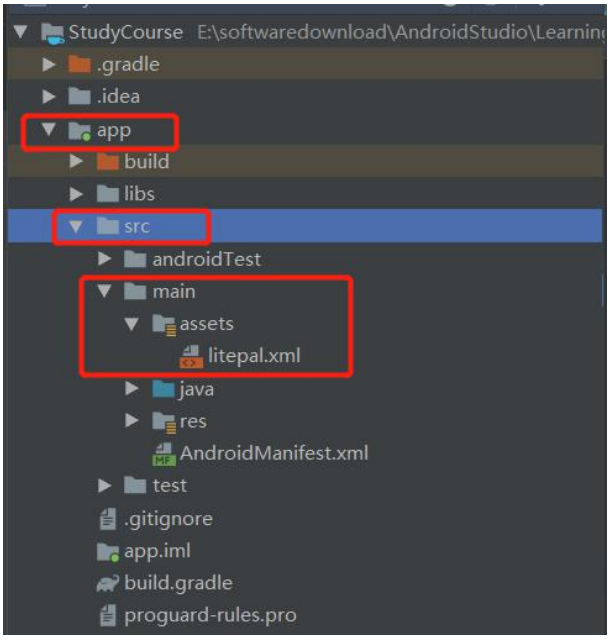
3) 配置 LitePal

a) 在(app/build.gradle)中添加 litepal 依赖

```
implementation 'org.litepal.android:core:1.4.1'
```

b) 创建 litepal.xml

在 app/src/main 目录创建 Directory，命名为"assets"，如下：



c) 在 assets 目录下创建"litepal.xml"文件，内容如下：

```
<?xml version="1.0" encoding="utf-8" ?>
<litepal>
    <dbname value="litePal_eLearning"></dbname>
    <version value="1"></version>
    <list>
        <mapping class="com.example.studycourse.Util.Course"></mapping>
    </list>
</litepal>
```

d) 配置 LitePalApplication，修改 AndroidManifest.xml

```
    <application
        android:name="org.litepal.LitePalApplication"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="E-Learning"
        android:roundIcon="@mipmap/cover"
        android:supportsRtl="true"
        android:theme="@style/AppTheme"
        android:usesCleartextTraffic="true">
```

4) 创建表实体

继承 DataSupport，并为每个属性添加 getter 和 setter

```
public class Course extends DataSupport {
    private int imageId;
    private String courseID;
    private String courseName;
    private String teacher;
    private String school;
    private String courseDate;
    private String courseDescription;
    private String imageURL;
    private int type;
```

5) 将实体类添加到“litepa.xml”映射中

```
<?xml version="1.0" encoding="utf-8" ?>
<litepal>
    <dbname value="litePal_eLearning"></dbname>
    <version value="1"></version>
    <list>
        <mapping class="com.example.studycourse.Util.Course"></mapping>
    </list>
</litepal>
```

6) 创建数据库

```
//初始化数据库
LitePal.getDatabase();
```

7) 在进入课程列表页面时，通过遍历数据库，查看数据库中是否已经存在已请求的课程信息，如果存在，则直接读取，如果不存在，则向服务器发送请求。

```
//判断数据库中是否已有课程信息，有则直接读取，没有则从网络请求
List<Course> courses = DataSupport.findAll(Course.class);
if(courses.isEmpty()){
    Log.d(TAG, msg: "getData: ");
    String address = "http://123.207.6.140:8080/getAllCourses";
    RequestBody requestBody = new FormBody.Builder().build();
    HttpUtil.sendOkHttpRequest(address, requestBody, new okhttp3.Callback() {
        @Override
        public void onFailure(Call call, IOException e) {

        }

        @Override
        public void onResponse(Call call, Response response) throws IOException {
            String responseData = response.body().string();
            showResponse(responseData);
        }
    });
}else{
    RecyclerviewAdapter reAdapter = new RecyclerviewAdapter( context: MainActivity.this,courses);
    recyclerView.setAdapter(reAdapter);
}
```

8) 当程序向服务器请求得到课程信息后，将其存入到数据库中。

```
Log.d(TAG, msg: "run: ");
JSONArray jsonArray = JSONArray.parseArray(response);
if(jsonArray.size()>0){
    for(int i=0;i<jsonArray.size();i++){
        JSONObject json = jsonArray.getJSONObject(i);
        String courseID = json.getString( key: "id");
        String name = json.getString( key: "name");
        String teacher = json.getString( key: "teacher");
        String school = json.getString( key: "school");
        String imageUrl = json.getString( key: "imageUrl");
        String videoUrl = json.getString( key: "videoUrl");
        String audioUrl = json.getString( key: "audioUrl");
        int type;
        if(videoUrl!=null&&audioUrl!=null){
            type = 1;
        }else {
            type = 0;
        }

        Course course = new Course(courseID,name,teacher,school,imageUrl,type);
        course.save();
        courseList.add(course);
    }
}
RecyclerviewAdapter reAdapter = new RecyclerviewAdapter( context: MainActivity.this,courseList);
recyclerView.setAdapter(reAdapter);
```

一、 定时课程广告

根据 Assignment 3 的第 6 点，要求使用广播（Broadcast）和通知（Notifications）实现提醒用户课程更新功能。本小组经讨论后决定通过修改本小组在 Assignment 2 中已经实现的课程信息推送，推广为课程广告功能。其逻辑如下：

用户进入登录界面后，通过开启线程来向服务器发送 HTTP 请求（接口：http://123.207.6.140:8080/getCourseAdvertisement）获取课程广告信息；请求响应成功后，发送广播（携带课程信息）；广播接收器获取广播内容后，生成一条通知并推送。本项目为测试方便，特将请求广告课程的时间间隔设为 10 秒。

关键代码：

- 请求线程（包含广播发送端）

```
new Thread((Runnable) () → {
        String address = "http://123.207.6.140:8080/getCourseAdvertisement";
        while(true){
            HttpUtil.sendOkHttpRequest(address, new okhttp3.Callback() {
                @Override
                public void onFailure(Call call, IOException e) {

                }

                @Override
                public void onResponse(Call call, Response response) throws IOException {
                    String responseData = response.body().string();
                    Intent intent = new Intent( action: "com.example.broadcast.LESSON_ADVERTISEMENT");
                    intent.putExtra( name: "data", responseData);
                    localBroadcastManager.sendBroadcast(intent);
                    Log.d(TAG, msg: "onResponse: 发送广播");
                }
            });

            try {
                Thread.sleep( millis: 10 * 1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
}).start();
```

- 广播接收器

```
class LocalReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent){
        String data = intent.getStringExtra( name: "data");
        Log.d(TAG, msg: "onReceive: 接收广播");
        try {
            JSONObject jsonObject = new JSONObject(data);

            String courseName = jsonObject.getString( name: "name");
            String courseTeacher = jsonObject.getString( name: "teacher");
            String courseSchool = jsonObject.getString( name: "school");
            String text = courseSchool + "新开课程：" + courseName + ", 专业名师 " + courseTeacher
                    + " 领衔主讲。";
            Log.d(TAG, msg: "onReceive: text:" + text);
            Notification notification = new NotificationCompat.Builder(context, channelId: "subscribe")
                    .setContentTitle("E-Learning 课程提醒")
                    .setContentText(text)
                    .setWhen(System.currentTimeMillis())
                    .setSmallIcon(R.drawable.audio)
                    .setLargeIcon(BitmapFactory.decodeResource(getResources(), R.drawable.audio))
                    .build();
            NotificationManager manager = (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
            manager.notify( id: 1, notification);
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
}
```