|  |
| --- |
| xmcu_logo |
| 移动应用开发实训 |
| 实训指导书 |
| 厦门城市职业学院 |
| 云计算技术与应用教研室 |
| 2015年9月修订 |

# 实训目的与要求

## 1.1 实训目的

1）掌握移动APP项目开发的流程；

2）掌握Android开发常用的工具软件；

3）掌握Android常用视图组件的使用；

4）掌握移动应用UI设计；

5）掌握Android四大组件的使用；

6）掌握网络通信技术；

7）掌握数据状态保存及调试技术；

8）初步掌握APP软件的压力测试方法；

## 1.2 实训任务

利用Android集成工具开发一个基于Android的APP项目——智慧城市；通过该项目的部分实现以熟悉移动应用项目开发的基本流程，从而检验并提高学生综合运用所学知识解决实际问题的能力。

## 1.3 实训环境

1）高性能PC（内存8G以上），电子教室及局域网；

2）Tomcat9.0以上，Mysql5.1, Android Studio, Android SDK；

# 二、实训内容及步骤

## 2.1 服务端环境安装及应用部署

安装【智慧城市】服务端应用。

Step1：安装MySql服务；

Step2：安装JDK8.0+服务;

Step3：安装Tomcat9.0服务；

Step4：安装smartcity.sql脚本

Step5：Tomcat安装目录中下的webapps是应用部署目录，将该目录下的ROOT目录更名为ROOT-tomcat，将【智慧城市】服务端应用目录ROOT拷贝到webapps目录下。

Step6：编辑ROOT\META-INF\context.xml文件，修改username="root"及 password="000000"以满足登录MySql服务。

Step7：测试。启动Tomcat9服务，打开浏览器，输入http://localhost:8080/prod-api/api/service/list，显示以下信息表示【智慧城市】服务端应用部署完成。



## 2.2 postman的简单使用及AS插件安装

### 2.2.1 安装postman及使用简介

参考https://blog.csdn.net/DataBaker/article/details/122127165 自行学习postman安装及使用。

参考https://blog.csdn.net/weixin\_42311968/article/details/122114218 自行学习postman的post请求。

### 2.2.2 GET/POST请求

1、启动【智慧城市】服务端应用，用postman实现GET请求。Url：http://localhost:8080/prod-api/api/service/list。返回结果用json格式显示。

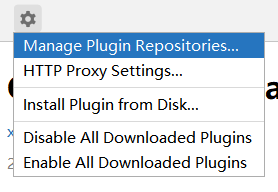
2、停止【智慧城市】服务端应用，运行androidtest.exe开启一个新的后端Web应用，用postman实现POST请求。Url：http://localhost:8080/testpost。Body内容格式采用x-www-form-urlencoded，参数为username、password。返回结果json格式显示。

### 2.2.3 GsonOrXmlFormat插件

* 下载：https://plugins.jetbrains.com/plugin/8435-gsonorxmlformat
* 安装：

1. Open your IDE and press Ctrl+Alt+S to open the IDE settings.

2. Select Plugins, click  and then click Install Plugin from Disk.

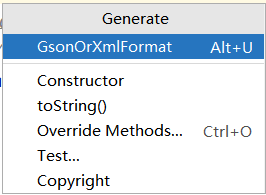


3. Select the plugin archive file and click OK.

4. Click OK to apply the changes and restart your IDE if prompted.

* 使用简介

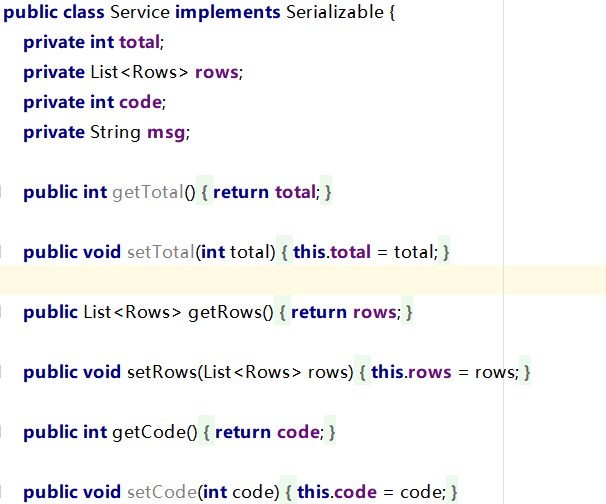
1. 先定义一个空类
2. 【Code】->【Generate...】->



3）<GsonOrXmlFormat插件使用.wmv>

### 2.2.4 GsonOrXmlForm应用

启动【智慧城市】服务端应用，用postman实现GET请求。Url：http://localhost:8080/prod-api/api/service/list，返回json格式数据，用该数据接合GsonOrXmlForm插件自动生成Service类的内容。如下所示：



## 2.3 项目功能说明

### 2.3.1 功能简介

智慧城市是一个比较大的系统工程，涉及领域众多，但在终极目标上简单明确，那就是服务民众，让民众的城市生活更美好。在便民惠民服务中，可以看到医疗、教育、金融、交通等领域成为热门。

搭建智慧城市的便民服务平台不仅要利用最先进的互联网线上手段，做好线上渠道，还要利用好传统的线下渠道，实现多渠道、广覆盖。如今提到一站式的服务平台，首先想到的就是各种移动客户端，利用线上线下更好的服务大众。便民服务平台作为智慧城市在产品层面的一个重要体现，成为一个大众参与的产品是其发展趋势和出路。

任务要求：

实现基于给定便民服务的新闻资讯、服务及个人中心等3个功能需求。

1、新闻资讯

* 获取所有新闻列表
* 获取推荐的新闻
* 按新闻类型获取新闻列表
* 获取新闻类型
* 发表新闻评论
* 获取新闻评论列表

新闻查询

1. 服务

* 获取服务信息

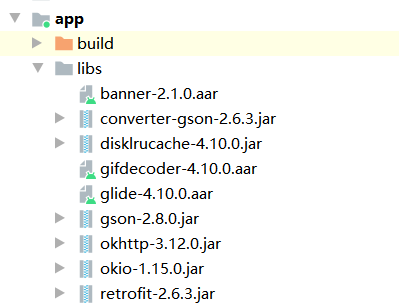
1. 个人中心

* 用户登录
* 获取用户信息
* 修改用户信息

### 2.3.2 创建工程SmartCity

Step1：用Android Studio创建名为SmartCity的工程。

Step2：拷贝工程需要的的依赖包至app/libs目录。



Step3：在app/build.gradle中添加：

implementation fileTree(dir: 'libs', include: ['\*.aar', '\*.jar'], exclude: [])

Step4：按工具条按钮  使依赖应用于工程。

Step5：添加权限及设置工程属性

1）添加字符串资源app\_name，其值为“智慧城市”

2）修改AndroidManifest.xml如下：

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="...">

**<uses-permission android:name="android.permission.INTERNET" />**

**<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />**

<application

...

android:label="@string/app\_name"

**android:usesCleartextTraffic="true"**

... >

...

</application>

</manifest>

## 2.4 实现引导页

### **2.4.1 实现目标**

app启动，页面出现第一张引导图，手指向左滑动，出现第二张引导图，依次操作到最后一张引导图（共计5张引导图），在最后一张图中添加两个按钮【进入主页】、【网络设置】，点击【进入主页】进入MainActivity。

### 2.4.2 在工程中添加依赖

如果2.3已经添加了banner-2.1.0.aar依赖，可忽略以下操作。

Step1：拷贝banner-2.1.0.aar到app/libs目录。

Step2：在app/build.gradle中添加：

implementation fileTree(dir: 'libs', include: ['\*.aar', '\*.jar'], exclude: [])

Step3：按工具条按钮  使依赖应用于工程。

### 2.4.3 自学轮播组件

学习地址：https://github.com/youth5201314/banner

或

<Android广告图片轮播控件基于ViewPager2实现.mhtml>

### 2.4.4 实现步骤

Step1：创建引导页GuideActivity。

Step2：修改AndroidManifest.xml，设置GuideActivity作为启动Activity。

Step3：将5张轮播图拷贝到res/drawable目录。

Step4：在GuideActivity的布局文件中添加Banner视图组件，可以设置以下属性

app:banner\_auto\_loop="false"

app:banner\_indicator\_selected\_color="@color/purple\_200"

app:banner\_infinite\_loop="false"

Step5：在res/layout中创建BannerAdapter对应的布局文件“banner\_image.xml”。

<?xml version="1.0" encoding="utf-8"?>

<ImageView xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/image"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:scaleType="centerCrop"/>

Step6：创建名为“GuideAdapter”的BannerAdapter

public class GuideAdapter extends BannerAdapter<Integer, GuideAdapter.ImageHolder> {

public GuideAdapter(List<Integer> data) {

super(data);

}

...

static class ImageHolder extends RecyclerView.ViewHolder {

...

}

}

Step7：在GuideActivity中添加实现逻辑代码。

* 实现主要步骤

1）定义数据源（图片资源），通常是一个List<Integer>，Integer是图片资源ID。

2）在onCreate方法实现去除app标题及全屏设置。

protected void onCreate(Bundle savedInstanceState) {

**requestWindowFeature(Window.FEATURE\_NO\_TITLE); getWindow().setFlags(WindowManager.LayoutParams.FLAG\_FULLSCREEN,**

**WindowManager.LayoutParams.FLAG\_FULLSCREEN);**

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_guide);

...

}

3）轮播组件初始化

1. 添加viewpager2的滑动监听
2. 给banner添加生命周期观察者，内部自动管理banner的生命周期
3. 设置banner轮播指示器 setIndicator(new CircleIndicator(this))
4. 设置指示器位置（左，中，右）
5. 添加viewpager的切换效果

4）创建BannerAdapter实例，注意数据源要作为参数。

5）设置BannerAdapter实例。

### 2.4.5 进阶

要求：

1. 在第一张轮播图添加【跳过】按钮，点击该按钮，关闭该GuideActivity，进入MainActivity。
2. 在最后一张轮播图中添加两个按钮【进入主页】、【网络设置】。

思路：

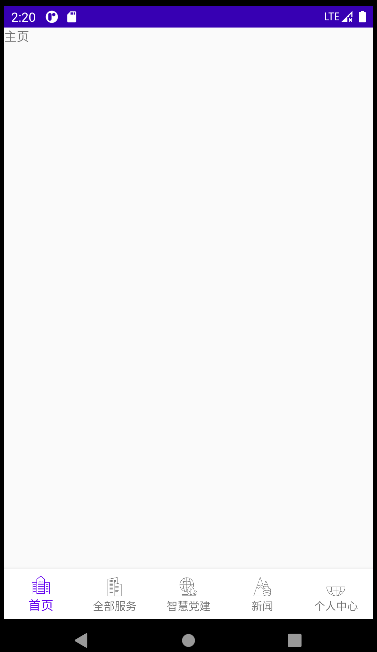
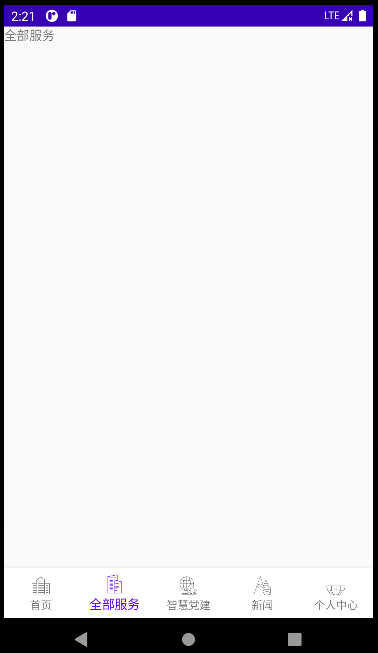
1）初始将【进入主页】、【网络设置】隐藏。

2）监听banner的viewpager2的滑动事件，判断为最后一张轮播图，现显示这两个按钮

3）点击【进入主页】，关闭该GuideActivity且进入MainActivity。

## 2.5 主页面框架设计

### 2.5.1 实现目标

显示底部导航栏，采用图标加文字方式显示，图标在上，文字在下，共五个图标分别为首页、全部服务、智慧党建、新闻、个人中心，点击标签进入对应页面。5个导航图标按钮，对应5个不同的页面，这些页面由Fragment构成，分别为：首页HomeFragment，全部服务ServiceFragment，智慧党建PartyconstructFragment，新闻NewsFragment，个人中心PersonalCenterFragment。

### 2.5.2 实现步骤

参照讲义【05 资源及Fragment】中的【NavigationDemo】案例。

Step1：创建导航菜单

Step2：在主办面引入BottomNavigationView组件，设置该组件的导航菜单

Step3：修改app的主题，将原有主题的parent换成

<style name="Theme...." parent="**Theme.AppCompat.Light.NoActionBar**">

目的是去除标题栏。

Step4：创建5个Fragment对象，将其置于Map表中。

Step5：设置BottomNavigationView组件的OnNavigationItemSelectedListener监听器，监听导航菜单项的状态，用FragmentManager实现Fragment对象的隐藏及显示以表达页面切换的效果。

## 2.6 首页设计

### 2.6.1 实现目标

1、主页显示搜索输入框，输入搜索内容后，点击软键盘“搜索”按钮，搜索新闻列表标题字段，将搜索到的新闻显示在RecycleView视图列表中。

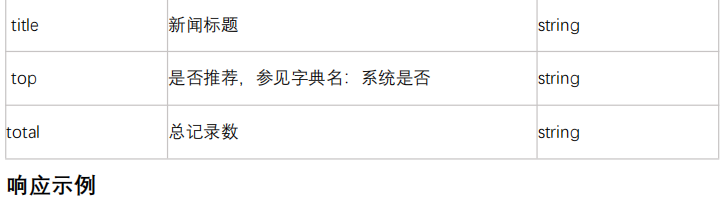
2、主页显示轮播图，点击轮播图以获取广告数据的targetId，用Toast显示该targetId。

### 2.6.2 API接口说明

#### 获取新闻列表







{

"msg": "success",

"total": 51,

"code": 200,

"rows": [

{

"publishDate": "2021-05-06",

"updateTime": "2021-05-10 09:20:12",

"remark": null,

"title": "六城市成智慧城市基础设施与智能网联汽车协同发展试点",

"type": 22,

"hot": "Y",

"params": {},

"content": "<p class=\"ql-align-justify\">据住建部网站消息，近...</p>",

"tags": "0",

"likeNum": 699,

"cover": "/upload/2bfc61d3-5d6e-4157-ba5f-a9dfe23af07f.jpg",

"commentNum": 2,

"createBy": null,

"subTitle": "0",

"top": "Y",

"readNum": 1232,

"createTime": "2020-10-23 16:17:56",

"updateBy": null,

"appType": "smart\_city",

"id": 8,

"searchValue": null,

"status": "Y"

},

{

...

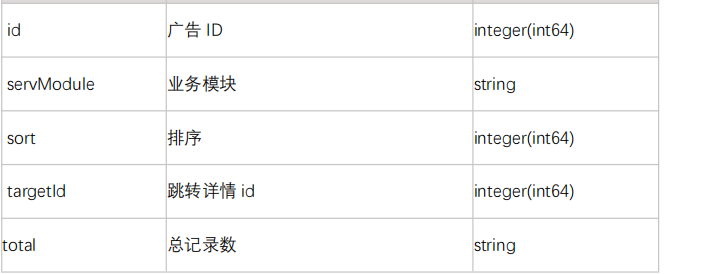
}

]

}

#### 查询引导页及主页轮播





响应示例

{

"msg": "success",

"total": 4,

"code": 200,

"rows": [

{

"servModule": "",

"targetId": 0,

"appType": "smart\_city",

"id": 24,

"sort": 1,

"advTitle": "开屏广告",

"type": 1,

"advImg": "/upload/d2aeef1a-7c47-46bc-8534-20b3d14cd8eb.png",

"status": 1

},

{

"servModule": "新闻详情",

"targetId": 28,

"appType": "smart\_city",

"id": 25,

"sort": 2,

"advTitle": "首页轮播",

"type": 2,

"advImg": "/upload/b9d9f081-8a76-41dc-8199-23bcb3a64fcc.png",

"status": 1

},

{

"servModule": "新闻详情",

"targetId": 29,

"appType": "smart\_city",

"id": 26,

"sort": 3,

"advTitle": "首页轮播",

"type": 2,

"advImg": "/upload/e614cb7f-63b0-4cda-bf47-db286ea1b074.png",

"status": 1

},

{

"servModule": "新闻详情",

"targetId": 30,

"appType": "smart\_city",

"id": 27,

"sort": 4,

"advTitle": "首页轮播",

"type": 2,

"advImg": "/upload/242e06f7-9fb0-4e16-b197-206f999c98f2.png",

"status": 1

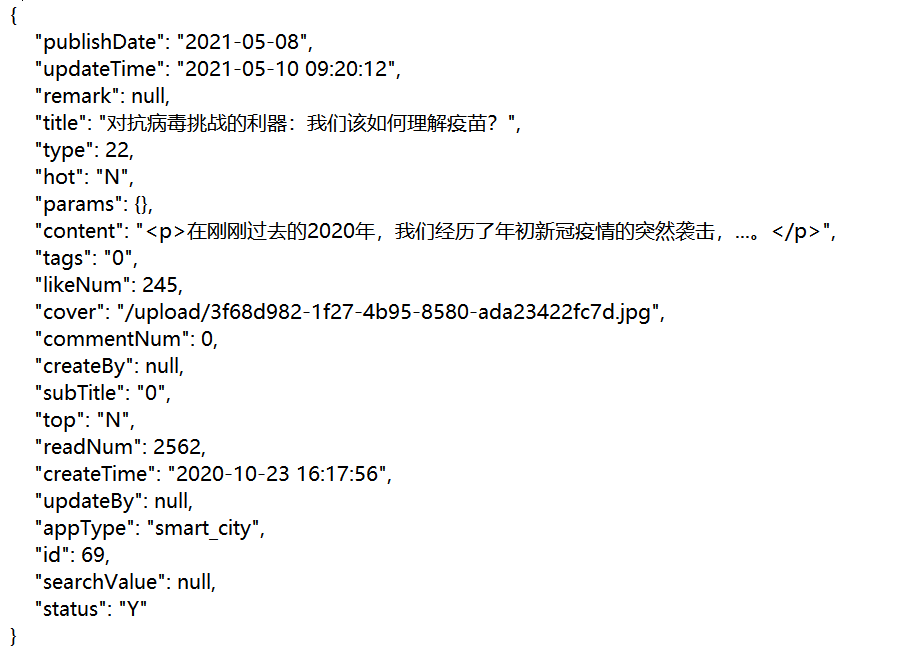
}

]

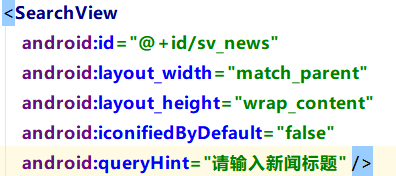
}

### 2.6.3 新闻资讯搜索

新闻资讯搜索数据来自于【获取新闻列表 接口地址 GET /prod-api/press/press/list】数据样例如下:



Step1：在主页HomeFragment对应的布局组件中添加SearchView：



Step2：在SearchView下再添加RecyclerView视图组件，以显示去搜索到的新闻标题列表。

Step3：自学SearchView视图组件用法。

学习网址：

https://www.jianshu.com/p/00cb87a2964f

或

https://blog.csdn.net/weixin\_46157140/article/details/108213511

将其中的ListView换成RecyclerView。

Step5：设计http请求的服务接口ApiService，方法getNews()。可参看讲义【07 线程及http】中【HttpDemo】案例。

Step6：设计新闻信息类News。具体字段参看postmain中的JOSN结果。

Step7：设计RecyclerView适配器。

Step8：RecyclerView设置RecyclerView适配器。

Step9：用Retrofit和OkHttp组件调用ApiService接口的getNews()方法获取新闻数据，将其置列表中。

Step10：设置SearchView的OnQueryTextListener监听器，在监听器方法onQueryTextSubmit中获取用户输入的新闻标题关键字，用该关键字在已经获取新闻数据列表中查找，将查找到的数据显示中RecyclerView视图中。

### 2.6.4 广告轮播

Step1：在主页HomeFragment对应的布局组件中添加轮播组件：

<com.youth.banner.Banner

android:id="@+id/banner"

android:layout\_width="match\_parent"

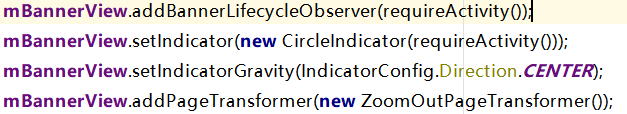
android:layout\_height="200dp"

app:banner\_indicator\_selected\_color="@color/purple\_200"/>

Step2：在接口ApiService定义方法getRotationData()。

Step3：设计新闻信息类RotationData。具体字段参看postmain中的JOSN结果。

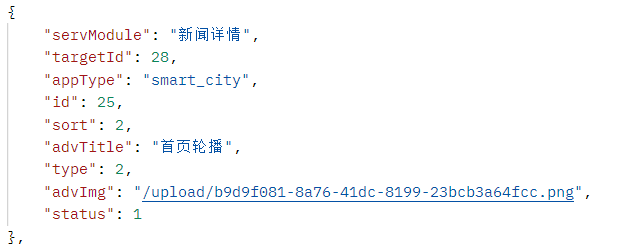
Step4：初始化轮播组件。



Step5：设计轮播组件适配器。

注意：

1. 网络图片资源用Glide获取。
2. 在网络图片上用setTag方法设置获取新闻targetId，如下图：



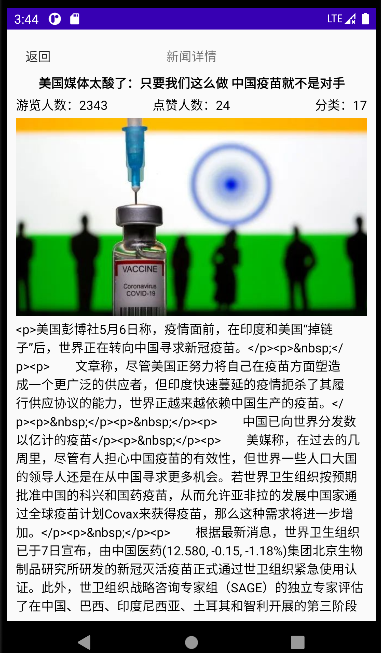
在图片的ImageView点击事件中用getTag获取targetId，用Toast显示。

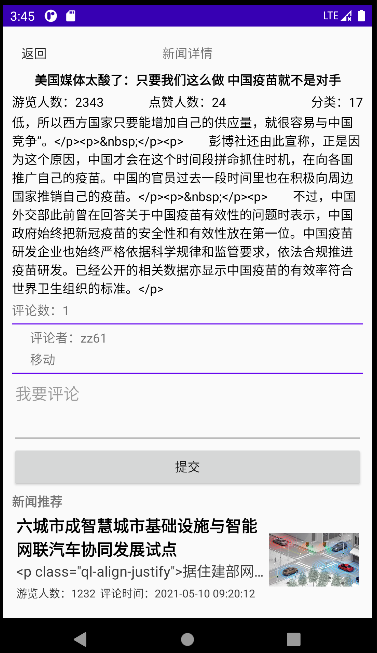
Step6：用Retrofit和OkHttp组件调用ApiService接口的getRotationData()方法获取广告数据，将其置列表中。

Step7：轮播组件设置轮播适配器。

## 2.7 新闻页设计

### 2.7.1 实现目标



1）新闻页面包含：轮播图、新闻分类、新闻列表等内容。轮播图可跳转新闻详情页面；新闻分类展示各类新闻主题；新闻列表根据最新发布时间排序，列表显示新闻图片、新闻名称、观看人数、点赞数等。

2）在新闻列表页面点击新闻名称跳转到新闻详情页面，信息如下：

①详情页面顶部栏显示新闻名称，点击“返回”按钮，返回上级目录。

②新闻详情内容按照（图片+文字）的形式进行展示，详情页面具有评论和查看评论列表功能，评论列表显示评论条数以及评论内容，用户可以对该新闻进行评论。

③详情页面还包括新闻推荐，以列表形式展示1-3篇推荐新闻，显示新闻名称、观看数、图片等信息。

### 2.7.2 API接口说明

详细参考【智慧城市项目 API 说明文档 V2.0】以下接口说明：

1.5.1.获取新闻分类 接口地址 GET /prod-api/press/category/list

1.5.2.获取新闻评论列表 接口地址 GET /prod-api/press/comments/list

1.5.3.获取新闻评论详细信息 接口地址 GET /prod-api/press/comments/{id}

1.5.4.新闻点赞 接口地址 PUT /prod-api/press/press/like/{id}

1.5.5.获取新闻列表 接口地址 GET /prod-api/press/press/list

1.5.6.获取新闻详细信息 接口地址 GET /prod-api/press/press/{id}

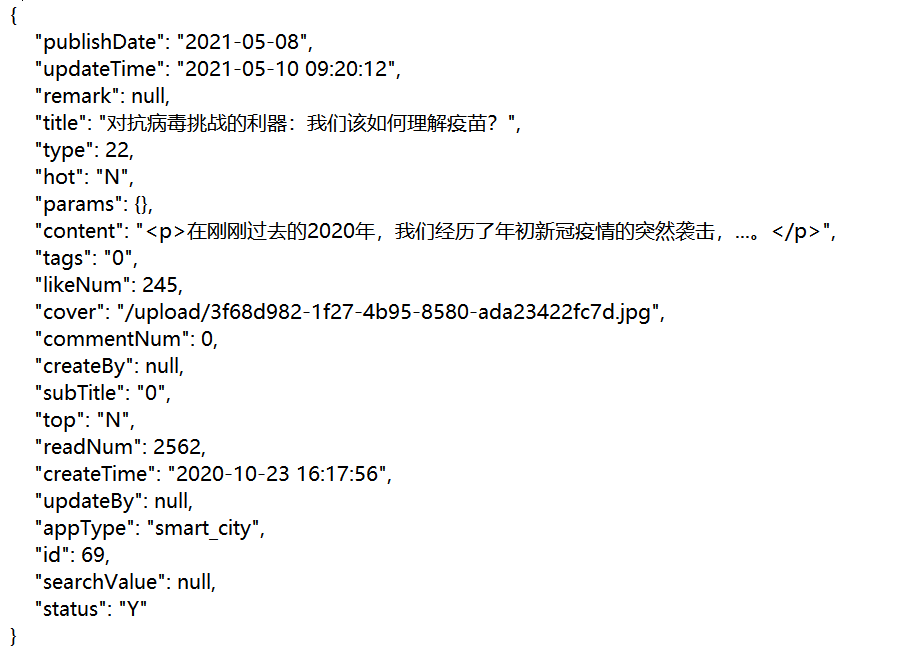
1.5.7.发表新闻评论 接口地址 POST /prod-api/press/pressComment

1.5.8.评论点赞 接口地址 PUT /prod-api/press/pressComment/like/{id}

### 2.7.3 新闻轮播图

新闻页面由NewsFragment实现。参看【2.5主页框架设计】。

轮播图数据来自于【获取新闻列表 接口地址 GET /prod-api/press/press/list】数据样例如下:



Step1：在NewsFragment对就的layout资源中进行界面设计，界面形式参看实现目标。

Step2：新闻数据获取参看2.6.3。

Step3：初始化轮播组件。

Step4：用Retrofit和OkHttp组件调用ApiService接口的getNews()方法获取新闻列表数据，将其置于List列表中。

Step5：设计轮播组件适配器。

注意：

1）新闻图片资源用Glide获取，置于ImageView中。

2）创建一个显示详细新闻内容的NewsDetailActivity，在onBindView方法中将新闻data用Intent传递给NewsDetailActivity。

1. 在NewsDetailActivity中显示新闻详情。

### 2.7.4 新闻详情

Step1：创建显示详细新闻内容的NewsDetailActivity

Step2：设计界面。

注意：

1. 新闻内容用WebView组件的loadData()方法，具体用法参照API，或Baidu。
2. 新闻详情内容按照（图片+文字）的形式进行展示
3. 详情页面顶部栏显示新闻名称，点击“返回”按钮，返回上级目录

参考界面如下：



Step3：从Intent中获取新闻数据，将其显示在布局组件中。

### 2.7.5 新闻评论

Step1：创建建新闻评论类NewsComment。字段内容查看API文档。

Step2：在接口ApiService中创建新闻评论/prod-api/press/comments/list" 方法getNewComment()。

注意：该方法有查询参数newsId。

Step3：NewsDetailActivity界面中添加RecycleView组件。

Step4：NewsDetailActivity界面中添加RecycleView组件。

Step5：用Retrofit和OkHttp组件调用ApiService接口的getNewComment()方法获取数据，将其置于List列表中。

Step6：设计RecycleView适配器。

Step7：RecycleView设置适配器。

### 2.7.6 Tab页布局

TabLaout与ViewPager2两个组件的结合可以为页面布局带来内容丰富的选择，类似于窗口控件的Tab选项卡：

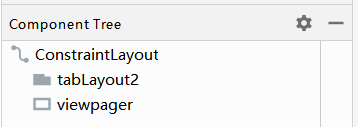
 

选项卡的标头由TabLaout实现，选项卡的内容页由Fragment实现，由ViewPager2控制。

* 实现步骤

Step1：创建一个新工程TablayoutViewpager

Step2：主界面设计



TabLayout，ViewPager2组件在Containers面板中。

* + res/layout/activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<com.google.android.material.tabs.TabLayout

android:id="@+id/tabLayout2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent">

</com.google.android.material.tabs.TabLayout>

<androidx.viewpager2.widget.ViewPager2

android:id="@+id/viewpager"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintHorizontal\_bias="0.0"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/tabLayout2" />

</androidx.constraintlayout.widget.ConstraintLayout>

Step3：Tab内容页界面

* res/layout/fragment\_tab.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:gravity="center\_horizontal|center\_vertical"

android:orientation="vertical">

<TextView

android:id="@+id/textView1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="200dp"

android:text="" />

</LinearLayout>

Step4：创建Fragment——Tab内容页

* MyFragment.java

public class MyFragment extends Fragment {

private int pageIndex;

List<String> mData;

private TextView textView;

public MyFragment(int pageIndex, List<String> data){

this.pageIndex = pageIndex;

this.mData = data;

}

@Nullable

@Override

public View onCreateView(@NonNull LayoutInflater inflater,

@Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

View rootView = inflater.inflate(R.layout.fragment\_tab, container, false);

textView = rootView.findViewById(R.id.textView1);

textView.setText("这是"+mData.get(pageIndex));

return rootView;

}

public static MyFragment newInstance(int pageIndex, List<String> data) {

MyFragment fragment = new MyFragment(pageIndex, data);

return fragment;

}

}

Step5：代码实现

* MainAcitivity.java

public class MainActivity extends AppCompatActivity {

private List<String> mData = new ArrayList<>();

private TabLayout tabLayout ;

private ViewPager2 viewPager;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

tabLayout = this.findViewById(R.id.tabLayout2);

viewPager = this.findViewById(R.id.viewpager);

initData();

initViewpage();

}

private void initData(){

mData.add("一班");

mData.add("二班");

mData.add("三班");

}

private void initViewpage() {

viewPager.setAdapter(

new FragmentStateAdapter(getSupportFragmentManager(),getLifecycle()) {

@NonNull

@Override

public Fragment createFragment(int position) {

return MyFragment.newInstance(position, mData );

}

@Override

public int getItemCount() {

return mData.size();

}

});

//关联并应用ViewPage2和Tab

new TabLayoutMediator(tabLayout, viewPager,

new TabLayoutMediator.TabConfigurationStrategy() {

@Override

public void onConfigureTab(@NonNull TabLayout.Tab tab, int position) {

tab.setText(mData.get(position));

}

}).attach();

}

}

### 2.7.7 新闻分类及新闻列表

实现方法：先获取新闻分类数据，再根据新闻类型id查询新闻，新闻分类名称可作为Tab标题，各个Tab页显示不同类型的新闻，每个Tab页对应一个Fragment组件，在Fragment实现新闻列表的显示。

Step1：用postman工具根据获取新闻分类接口地址 GET /prod-api/press/category/list 获取新闻分类内容。



Step2：根据以上内容创建NewsCategory类

Step3：在接ApiService口中创建GET /prod-api/press/category/list 方法getNewsCategory()。

Step4：用Retrofit和OkHttp组件调用ApiService接口的getNewsCategory()方法获取数据，将其置于List列表中。

Step5：在轮播图下放置Tablayout及ViewPager2视图组件。

Step6：根据2.7.6方法实现Tab布局，Tab标题由NewsCategory的name字段填充。

Step7：在创建Tab页内容Framgent时，将NewsCategory的id传入，在Framgent中实现根据新闻类型ID获取新闻列表。

Step8：在创建Tab页内容Framgent时，将NewsCategory的id传入，在Framgent中实现根据新闻类型id、页面起始号1，页面大小10作为参数获取新闻列表，参数形式参看API 33页，参数类型是query。最终形式如下：

@GET("/prod-api/press/press/list")

Call<News> getNews(@Query("pageNum") int pageNum,

@Query("pageSize") int pageSize, @Query("type") int type);

Step9：新闻数据获取实现参看【2.6.3 新闻资讯搜索】

## 2.8 个人中心

### 2.8.1 实现目标

1）首先进入个人中心页面，个人中心页面显示用户头像、账户、个人信息页面入口、修改密码页面入口，订单列表页面入口、意见反馈页面入口，点击“退出”按钮可退出登录。

2）点击个人信息跳转至个人信息页面，标签栏显示本页面标题，点击返回图标可返回到上一页，点击修改可保存修改的信息，可修改内容为：头像、昵称、性别、联系电话，注：证件号只显示前两位与后四位数字其他使用 \* 号代替。

3）点击修改密码可进入修改密码页面，标签栏显示本页面标题，点击返回图标可返回到上一页，输入原密码与新密码，点击“确定”按钮可保存修改的信息。

图 2.8.1-1 图 2.8.1-2

图 2.8.1-3 图 2.8.1-4

### 2.8.2 API接口说明

详细参考【智慧城市项目 API 说明文档 V2.0】以下接口说明：

1.1.1.用户登录 接口地址 POST /prod-api/api/login

1.2.1.查询个人基本信息 接口地址 GET /prod-api/api/common/user/getInfo

1.2.2.修改个人基本信息 接口地址 PUT /prod-api/api/common/user

1.2.3.修改用户密码 接口地址 PUT /prod-api/api/common/user/resetPwd

### 2.8.3 用户登录

以下操作均在PersonalCenterFragment中实现。

Step1：根据图 2.8.1-1设计界面。

Step2：根据图 2.8.1-2创建用于登录的Activity及界面。

Step3：用postman工具根据POST /prod-api/api/login 获取登录成功后的返回内容。

Step4：根据返回内容创建LoginResult类，根据POST请求参数创建LoginUser类。

Step5：在接口ApiService中创建POST /prod-api/api/login 方法login()。

Step6：在【登录】按钮的点击事件中发出登录请求。

Step7：登录成功能后得到一个Token，用SharedPreferences保存到磁盘。销毁登录的Activity，并将Token返回给MainActivity，MainActivity在onActivityResult回调方法中再调用PersonalCenterFragment中的onActivityResult。

### 2.8.4 获取用户信息

Step1：在PersonalCenterFragment中的onActivityResult中实现用户信息的获取。具体如下：

Step2：用postman工具根据GET /prod-api/api/common/user/getInfo获取用户信息。注意要用刚才获取到的token作为头部认证信息，具体参看API。根据返回的数据创建UserInfo类。

Step3：在ApiService接口中创建GET /prod-api/api/common/user/getInfo方法getUserInfo()。

@GET("/prod-api/api/common/user/getInfo")

Call<UserInfo> getUserInfo(@Header("Authorization") String token);

Step4：用Retrofit和OkHttp组件调用ApiService接口的getUserInfo()方法获取数据，填充到图 2.8.1-1中的视图组件中，将获取到的用户数据通过SharedPreferences对象保存到磁盘。并设置【登录】按钮标题为【退出】。视图效果如图 2.8.1-3。

### 2.8.5 修改用户信息

Step1：创建修改用户信息的UserInfoActivity。

Step2：在ApiService接口中创建PUT /prod-api/api/common/user方法updateUserInfo()

@PUT("/prod-api/api/common/user")

Call<Result> updateUserInfo(@Header("Authorization") String token,

@Body UserInfo.User userInfo);

注意：Result类是操作的返回结果类，具体字段查询API。

Step3：在【个人信息】组件上添加点击事件，激活UserInfoActivity，将保存在磁盘中的用户信息填到 图 2.8.1-4中视图组件中。

Step4：点击【修改】按钮，在其点击事件方法中用Retrofit和OkHttp组件调用ApiService接口的updateUserInfo()方法修改数据，根据响应结果显示操作结果信息。

### 2.8.6 初始化用户信息

当用户切换到【个人中心】时，根据磁盘上是否保存用户信息作为判断条件，以判断用户是否已处理于登录状态，由此，初始成图 2.8.1-1（未登录状态）或图 2.8.1-3（已登录状态）。

### 2.8.7 退出登录

Step1：在图 2.8.1-3状态下，给【退出】按钮添加点击事件方法，在该方法中删除保存在磁盘上的用户信息。

Step2：清空与用户信息相关的视图组件数据。

Step3：将【退出】按钮标题改为【登录】。