**深 圳 大 学 实 验 报 告**

**课程名称： 移动设备交互应用**

**实验项目名称： 我的头条**

**学院： 计算机与软件学院**

**专业： 计算机科学与技术**

**指导教师： 解为成**

**报告人： 沈晨玙 学号：2019092121 班级： 19计科国际**

**实验时间： 2020.11.3**

**实验报告提交时间： 2020.11.5**

**教务处制**

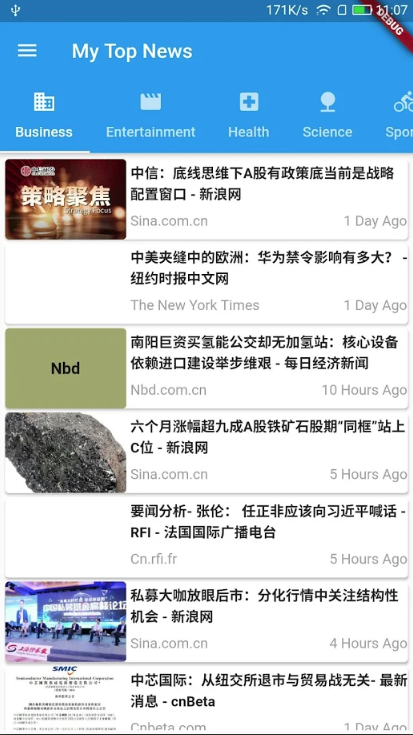
一、实验目的与内容：

**目的：**掌握安卓中活动的编写、自定义用户界面的开发、能使用HTTP协议访问网络；并能通过自学能适当完善该APP界面，并使界面尽量美观。

**内容要求：**

1. 请尽量模拟如下APP界面的功能，参考：

https://play.google.com/store/apps/details?id=mark.h.my\_news\_app&hl=en\_US



2. 该实现的界面应至少包含3个菜单，分别展示个人3个方面的信息，菜单之间要表现出一定的差异性；每个菜单可以包含2-5个条目，每个条目能响应个人某方面的偏好信息；此外，如果是响应网页，需同时体现出a) 采用浏览器浏览 与 b) 下载到本地 两种技术方案。

3. 尽量多的应用参考书《第一行代码 Android》第二版第2章（活动）、第3章（UI开发）与第9章（网络技术）的各个知识点。

注意：

1. 实验报告中需要有功能的描述、实验结果的截屏图像及详细说明；

2. 该实验报告的所需的部分内容需要自学（如第9章）；

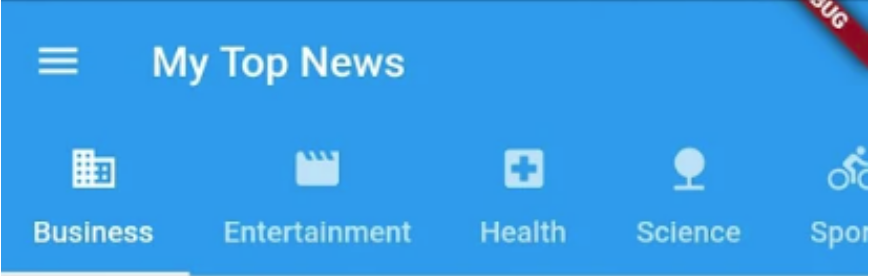
3. 也欢迎采用其它章节的知识点完成本次实验报告，如果实现的功能言之合理，会考虑酌情加分。

二、实验过程和代码与结果

**1.“我的头条”APP的构建过程及结果**

**1.1 标题栏设计**

**1.1.1 标题栏UI布局分析**



·上半部分左侧Button控件控制侧菜单栏，右侧是TextView控件显示软件标题

·下半部分为一个横向的RecyclerView，每一个组件由ImageView与TextView组成

**1.1.2 标题栏UI布局代码构建**

·创建drawable--xxhdpi文件夹，用于存放图片资源，利用自动匹配机制去选择对应的布局和图片资源（本项目矢量图来自阿里巴巴矢量图标库https://www.iconfont.cn/）

·在layout文件夹中，创建title.xml，编写布局代码

·标题栏为线性布局，背景颜色设置为浅蓝色，完成整体设计

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

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

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

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#17d9ff"

android:orientation="vertical">

<--上下部分代码-->

</LinearLayout>

上半部分：

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal"

android:paddingBottom="10dp">

<Button

android:layout\_width="30dp"

android:layout\_height="30dp"

android:id="@+id/sideMenuButton"

android:layout\_marginStart="15dp"

android:layout\_marginTop="8dp"

android:layout\_marginBottom="8dp"

android:layout\_marginEnd="30dp"

android:background="@drawable/sidemenu"/>

<TextView

android:id="@+id/appTitle"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:layout\_gravity="left"

android:text="@string/my\_top\_news"

android:textSize="30sp"

android:textColor="#ffffff"

tools:ignore="RtlHardcoded"

android:typeface="serif"/>

</LinearLayout>

<TextView

android:id="@+id/appTitle"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:layout\_gravity="left"

android:text="@string/my\_top\_news"

android:textSize="30sp"

android:textColor="#ffffff"

tools:ignore="RtlHardcoded"

android:typeface="serif"/>

</LinearLayout>

下半部分

·使用RecyclerView，由于属于新增控件，Google将RecyclerView定义在AndroidX中，需要在项目的build.gradle中添加RecyclerView依赖，保证在所有Android系统版本上都可以使用RecyclerView控件。

·打开app/build.gradle文件，在dependencies闭包中添加以下内容

dependencies {

implementation "org.jetbrains.kotlin:kotlin-stdlib:$kotlin\_version"

...

implementation 'androidx.recyclerview:recyclerview:1.1.0'

...

androidTestImplementation 'androidx.test.espresso:espresso-core:3.3.0'

}

·打开title.xml，编写布局代码

<androidx.recyclerview.widget.RecyclerView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@+id/recyclerViewForTitle"/>

·最终效果图



**1.1.3 主界面UI布局构建**

·因为标题栏在所有的界面都需要展示，所以利用<include layout="@layout/title"/>整体引入部分，简化代码

·主界面可以自行设计内容，此处为利用相对布局，居中展示个人信息

·打开activity\_main.xml文件，编写布局代码

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

<LinearLayout 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:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<include layout="@layout/title"/>

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_centerInParent="true"

android:id="@+id/name"

android:textSize="30sp"

android:text="姓名：沈晨玙"/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_above="@id/name"

android:layout\_centerInParent="true"

android:textSize="30sp"

android:text="班级：2019计科国际班"/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/name"

android:layout\_centerInParent="true"

android:textSize="30sp"

android:text="学号：2019092121"/>

</RelativeLayout>

</LinearLayout>

android:layout\_height="wrap\_content"

android:layout\_above="@id/name"

android:layout\_centerInParent="true"

android:textSize="30sp"

android:text="班级：2019计科国际班"/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/name"

android:layout\_centerInParent="true"

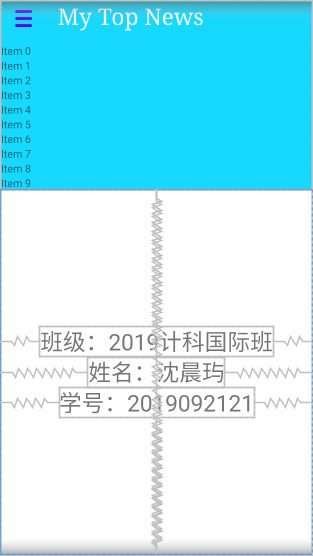
android:textSize="30sp"

android:text="学号：2019092121"/>

</RelativeLayout>

</LinearLayout>

·最终效果图



**1.2 标题栏RecyclerView代码编写**

·RecyclerView基本组成：

标题类：TitleMenu.kt

适配器：TitleMenuAdapter.kt

子项布局：titlemenu\_item.xml

**1.2.1 标题栏RecyclerView布局代码**

·创建子项布局：titlemenu\_item.xml

·线性布局，由ImageView与TextView组成。因为是RecyclerView要完成水平排列，item要改成垂直排列，所以LinearLayout改成垂直方向排列，并使用layout\_marginTop让文字与图片保持一定距离。

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

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

android:layout\_width="90dp"

android:orientation="vertical"

android:layout\_height="wrap\_content">

<ImageView

android:layout\_width="40dp"

android:layout\_height="40dp"

android:id="@+id/titleMenuImage"

android:layout\_gravity="center\_horizontal"

android:layout\_marginTop="10dp"/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/titleMenuName"

android:layout\_gravity="center\_horizontal"

android:layout\_marginTop="10dp"/>

</LinearLayout>

·最终效果图



**1.2.2 标题栏RecyclerView Kotlin代码**

①标题类：TitleMenu.kt

·创建TitleMenu.kt

·编写TitleMenu类的具体内容，包含name与imageID，代表栏目名与图片

package com.example.experiment2

class TitleMenu(val name: String, val imageID: Int)

②适配器：TitleMenuAdapter.kt

·创建TitleMenuAdapter.kt，传入参数为标题栏项目列表titleMenuList: List<TitleMenu>

class TitleMenuAdapter(val titleMenuList: List<TitleMenu>) :

RecyclerView.Adapter<TitleMenuAdapter.ViewHolder>() {

......

}

·编写TitleMenuAdapter.kt的具体内容，重写以下方法

·ViewHolder(view: View) : RecyclerView.ViewHolder(view)

·定义一个内部类ViewHolder，他要继承自RecyclerView.ViewHolder。然后ViewHolder的主构造函数中要传入一个View参数，这个参数通常就是RecyclerView子项的最外层布局，那么就可以通过findViewById（）方法来获取布局中ImageView和TextView的实例了。

inner class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {

val titleMenuImage: ImageView = view.findViewById(R.id.titleMenuImage)

val titleMenuName: TextView = view.findViewById(R.id.titleMenuName)

}

·onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder

·创建ViewHolder实例，将布局加载进来，然后创建一个ViewHolder实例，并把加载出来的布局传入构造函数，最后将ViewHolder的实例返回。

override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {

val view =

LayoutInflater.from(parent.context).inflate(R.layout.titlemenu\_item, parent, false)

//点击事件

val viewHolder = ViewHolder(view)

return viewHolder

}

·onBindViewHolder(holder: ViewHolder, position: Int)

·对RecyclerView子项的数据进行赋值，会在每个子项将滚到屏幕内的时候执行，通过position参数得到当前项的实例，然后将数据设置到ViewHolder的ImageView和TextView当中即可。

override fun onBindViewHolder(holder: ViewHolder, position: Int) {

val titleMenu = titleMenuList[position]

holder.titleMenuImage.setImageResource(titleMenu.imageID)

holder.titleMenuName.text = titleMenu.name

}

·getItemCount()

·告诉RecyclerView一共有多少子项，返回数据源的长度。

override fun getItemCount() = titleMenuList.size

③主活动：MainActivity.kt

·因为有自定义的标题栏，所以隐藏程序默认的标题栏。在onCreate方法中加入如下函数。

supportActionBar?.hide()

·使用initTitleMenu（）方法，用于初始化所有标题数据。接着在onCreate（）方法中创建了一个LinearLayoutManager用于指定RecylerView的布局方式，调用LinearLayoutManager的setOrientation（）方法设置布局的排列方向，默认是纵向排列，传入LinearLayoutManager.HORIZONTAL，表示让布局横行排列，这样就可以横向滚动了。然后创建TitleMenuAdapter的实例，并将标题数据传入TitleMenuAdapter的构造函数中，最后调用RecyclerView的setAdapter方法来完成适配器设置，这样RecyclerView和数据之间的关联就建立完成了。

class MainActivity : AppCompatActivity() {

private val titleMenu = ArrayList<TitleMenu>()

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

supportActionBar?.hide()

initTitleMenu()

val layoutManager = LinearLayoutManager(this)

val recyclerView: RecyclerView = findViewById(R.id.recyclerViewForTitle)

layoutManager.orientation=LinearLayoutManager.HORIZONTAL

recyclerView.layoutManager = layoutManager

val adapter = TitleMenuAdapter(titleMenu)

recyclerView.adapter = adapter

}

private fun initTitleMenu() {

titleMenu.add(TitleMenu("Business", R.drawable.business))

titleMenu.add(TitleMenu("Entertainment", R.drawable.entertainment))

titleMenu.add(TitleMenu("Health", R.drawable.health))

titleMenu.add(TitleMenu("Science", R.drawable.science))

titleMenu.add(TitleMenu("Sport", R.drawable.sport))

titleMenu.add(TitleMenu("Game", R.drawable.game))

}

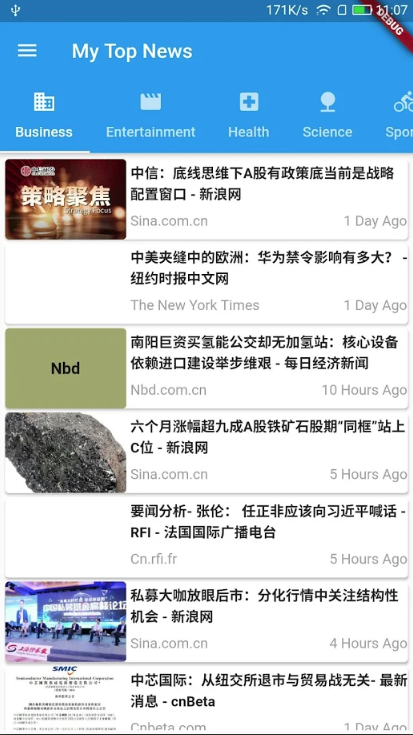
}

最终效果图：



**1.3 新闻栏设计**

**1.3.1 新闻栏UI布局分析**



·新闻栏整体为一个垂直的RecyclerView

·每一个组件由左侧的ImageView与右侧的线性布局组成，其中右侧又由三个TextView组成，其中分为线性与相对布局。最下方为一个View组件，颜色为纯灰色，用于分割新闻。

**1.3.2 新闻界面UI布局构建**

·因为标题栏在所有的界面都需要展示，所以利用<include layout="@layout/title"/>整体引入部分，简化代码

·创建新活动，以business界面为例，New->Activity->Empty Activity,创建Business.kt与activity\_business.xml文件，首先打开activity\_business.xml，编写布局代码。（其他界面同原理）

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

<LinearLayout 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"

android:orientation="vertical"

tools:context=".Business">

<include layout="@layout/title"/>

<androidx.recyclerview.widget.RecyclerView

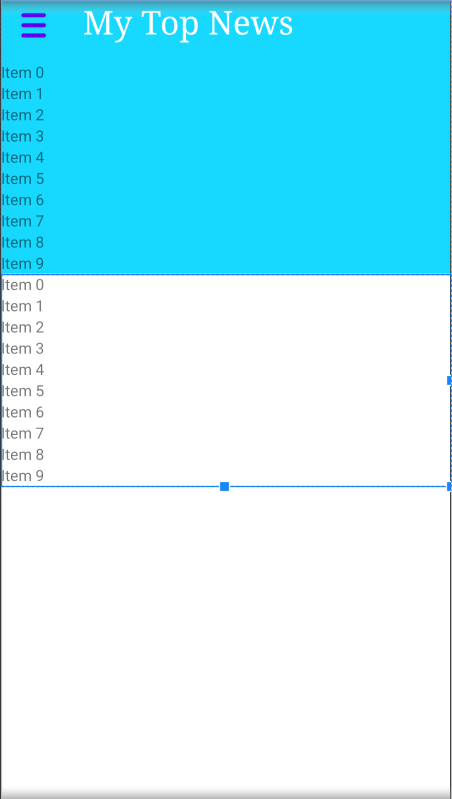
android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/recyclerViewForBusiness"/>

</LinearLayout>

最终效果图：



**1.4 新闻栏RecyclerView代码编写**

**·**·RecyclerView基本组成：

新闻类：vertical\_news.kt和flow\_news.kt

适配器：vertical\_newsAdapter.kt和flow\_newsAdapter.kt

子项布局：vertical\_news\_item.xml和flow\_news\_item.xml

**1.4.1 新闻栏RecyclerView布局代码**

·在layout文件夹中，创建vertical\_news\_item.xml，编写垂直布局代码

·每一个组件由左侧的ImageView与右侧的线性布局组成，其中右侧又由三个TextView组成，其中分为线性与相对布局。相对布局中利用alignParentEnd，alignParentStart，alignParentBottom属性确定左下角右下角属性。最下方为一个View组件，颜色为纯灰色，用于分割新闻。

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

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

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

android:layout\_width="match\_parent"

android:layout\_height="102dp"

android:orientation="vertical">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="100dp">

<ImageView

android:id="@+id/verticalNewsImage"

android:layout\_width="130dp"

android:layout\_height="100dp"

android:layout\_marginLeft="10dp" />

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="100dp"

android:orientation="vertical">

<TextView

android:id="@+id/verticalNewsTitle"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_gravity="top"

android:fontFamily="sans-serif-condensed"

android:gravity="left"

android:padding="5dp"

android:textSize="16sp"

tools:ignore="RtlHardcoded " />

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/verticalNewsURL"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:layout\_alignParentStart="true"

android:padding="5dp"

android:textSize="8sp"/>

<TextView

android:id="@+id/verticalNewsTime"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentEnd="true"

android:layout\_alignParentBottom="true"

android:textSize="8sp"

android:padding="5dp"/>

</RelativeLayout>

</LinearLayout>

</LinearLayout>

<View

android:layout\_width="match\_parent"

android:layout\_height="1dp"

android:background="#A3A0A0" />

</LinearLayout>

·最终效果图：



·在layout文件夹中，创建flow\_news\_item.xml，编写瀑布布局代码

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

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

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

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_margin="5dp"

android:orientation="vertical">

<ImageView

android:layout\_width="130dp"

android:layout\_height="100dp"

android:id="@+id/flowNewsImage"

android:layout\_gravity="center\_horizontal"

android:layout\_marginTop="10dp"/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_gravity="left"

android:id="@+id/flowNewsTitle"

android:layout\_marginTop="10dp"/>

</LinearLayout>

·最终效果图：



**1.4.2 新闻栏RecyclerView Kotlin代码**

·大致与标题栏RecyclerView代码功能一致，所以基本以贴代码为主。

①新闻类：vertical\_news.kt和flow\_news.kt

·创建vertical\_news.kt和flow\_news.kt

package com.example.experiment2

class vertical\_news(

val imageID: Int,

val newsTitle: String,

val newsURL: String,

val newsTime: String,

val URL: String,

val content: String

)

package com.example.experiment2

class flow\_news(

val imageID: Int,

val newsTitle: String,

val newsURL: String,

val newsTime: String,

val URL: String,

val content: String

)

②适配器：vertical\_newsAdapter.kt和flow\_newsAdapter.kt

·创建vertical\_newsAdapter.kt

package com.example.experiment2

import android.app.Activity

import android.app.AlertDialog

import android.content.Intent

import android.net.Uri

import android.view.LayoutInflater

import android.view.View

import android.view.ViewGroup

import android.widget.ImageView

import android.widget.TextView

import android.widget.Toast

import androidx.recyclerview.widget.RecyclerView

class vertical\_newsAdapter(val vertical\_newsList: List<vertical\_news>) :

RecyclerView.Adapter<vertical\_newsAdapter.ViewHolder>() {

inner class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {

val newsImage: ImageView = view.findViewById(R.id.verticalNewsImage)

val newsTitle: TextView = view.findViewById(R.id.verticalNewsTitle)

val newsURL: TextView = view.findViewById(R.id.verticalNewsURL)

val newsTime: TextView = view.findViewById(R.id.verticalNewsTime)

}

inner class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {

val newsImage: ImageView = view.findViewById(R.id.verticalNewsImage)

val newsTitle: TextView = view.findViewById(R.id.verticalNewsTitle)

val newsURL: TextView = view.findViewById(R.id.verticalNewsURL)

val newsTime: TextView = view.findViewById(R.id.verticalNewsTime)

}

override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {

val view = LayoutInflater.from(parent.context)

.inflate(R.layout.vertical\_news\_item, parent, false)

return viewHolder

}

override fun onBindViewHolder(holder: ViewHolder, position: Int) {

val verticle\_news = vertical\_newsList[position]

holder.newsImage.setImageResource(verticle\_news.imageID)

holder.newsTitle.text = verticle\_news.newsTitle

holder.newsTime.text = verticle\_news.newsTime

holder.newsURL.text = verticle\_news.newsURL

}

override fun getItemCount() = vertical\_newsList.size

}

·创建flow\_newsAdapter.kt

package com.example.experiment2

import android.app.Activity

import android.app.AlertDialog

import android.content.Intent

import android.net.Uri

import android.view.LayoutInflater

import android.view.View

import android.view.ViewGroup

import android.widget.ImageView

import android.widget.TextView

import android.widget.Toast

import androidx.recyclerview.widget.RecyclerView

class flow\_newsAdapter(val flow\_newsList: List<flow\_news>) :

RecyclerView.Adapter<flow\_newsAdapter.ViewHolder>() {

val newsImage: ImageView = view.findViewById(R.id.flowNewsImage)

val newsTitle: TextView = view.findViewById(R.id.flowNewsTitle)

}

inner class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {

val newsImage: ImageView = view.findViewById(R.id.flowNewsImage)

val newsTitle: TextView = view.findViewById(R.id.flowNewsTitle)

}

override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {

val view = LayoutInflater.from(parent.context)

.inflate(R.layout.flow\_news\_item, parent, false)

val viewHolder = ViewHolder(view)

return viewHolder

}

override fun onBindViewHolder(holder: ViewHolder, position: Int) {

val flow\_news = flow\_newsList[position]

holder.newsImage.setImageResource(flow\_news.imageID)

holder.newsTitle.text = flow\_news.newsTitle

}

override fun getItemCount() = flow\_newsList.size

}

③新闻界面活动（Business作为垂直布局例子，Game作为瀑布布局例子）

·编写Business.kt

·因为有自定义的标题栏，所以隐藏程序默认的标题栏，并引入标题栏RecyclerView Kotlin代码。在onCreate（）方法中创建了一个LinearLayoutManager用于指定新闻栏RecylerView的布局方式，默认是纵向排列。然后创建TitleMenuAdapter的实例，并将标题数据传入TitleMenuAdapter的构造函数中，最后调用RecyclerView的setAdapter方法来完成适配器设置，这样RecyclerView和数据之间的关联就建立完成了。

package com.example.experiment2

import androidx.appcompat.app.AppCompatActivity

import android.os.Bundle

import androidx.recyclerview.widget.LinearLayoutManager

import androidx.recyclerview.widget.RecyclerView

class Business : AppCompatActivity() {

private val verticalNews = ArrayList<vertical\_news>()

private val titleMenu = ArrayList<TitleMenu>()

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_business)

supportActionBar?.hide()

//菜单栏配置

initTitleMenu()

val layoutManager1 = LinearLayoutManager(this)

val recyclerView1: RecyclerView = findViewById(R.id.recyclerViewForTitle)

layoutManager1.orientation = LinearLayoutManager.HORIZONTAL

recyclerView1.layoutManager = layoutManager1

val adapter1 = TitleMenuAdapter(titleMenu)

recyclerView1.adapter = adapter1

//新闻栏配置

initVerticalNewsMenu()

val layoutManager = LinearLayoutManager(this)

val recyclerView: RecyclerView = findViewById(R.id.recyclerViewForBusiness)

recyclerView.layoutManager = layoutManager

val adapter = vertical\_newsAdapter(verticalNews)

recyclerView.adapter = adapter

}

private fun initVerticalNewsMenu() {

repeat(3){

verticalNews.add(

vertical\_news(

R.drawable.businessnew1,

"富士康的两难境地：美国工厂难产 中国市场竞争压力剧增",

"finance.sina.com.cn",

"2020年10月24日 07:13",

"https://finance.sina.com.cn/chanjing/gsnews/2020-10-24/doc-iiznezxr7762539.shtml",

" 曾经被美国总统特朗普誉....."

)

)

verticalNews.add(...)

verticalNews.add(...)

}

}

private fun initTitleMenu() {

titleMenu.add(TitleMenu("Business", R.drawable.business))

titleMenu.add(TitleMenu("Entertainment", R.drawable.entertainment))

titleMenu.add(TitleMenu("Health", R.drawable.health))

titleMenu.add(TitleMenu("Science", R.drawable.science))

titleMenu.add(TitleMenu("Sport", R.drawable.sport))

titleMenu.add(TitleMenu("Game", R.drawable.game))

}

}

titleMenu.add(TitleMenu("Health", R.drawable.health))

titleMenu.add(TitleMenu("Science", R.drawable.science))

titleMenu.add(TitleMenu("Sport", R.drawable.sport))

titleMenu.add(TitleMenu("Game", R.drawable.game))

}

}

·编写Game.kt

·大部分部分与上方代码相同。不同点在于瀑布布局的编写。在onCreate（）方法中，创建了一个StaggeredGridLayoutManager的实例。StaggeredGridLayoutManager的构造函数接受两个参数：第一个参数用于指定布局的列数，传入3表示会把布局分为3列；第二个参数用于指定布局的排列方式，传入StaggeredGridLayoutManager.VERTICAL表示会让布局纵向排列。最后把创建好的实例设置到RecyclerView当中就可以了。

package com.example.experiment2

import androidx.appcompat.app.AppCompatActivity

import android.os.Bundle

import androidx.recyclerview.widget.LinearLayoutManager

import androidx.recyclerview.widget.RecyclerView

import androidx.recyclerview.widget.StaggeredGridLayoutManager

class Game : AppCompatActivity() {

private val flowNews = ArrayList<flow\_news>()

private val titleMenu = ArrayList<TitleMenu>()

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_game)

supportActionBar?.hide()

//菜单栏配置

initTitleMenu()

val layoutManager1 = LinearLayoutManager(this)

val recyclerView1: RecyclerView = findViewById(R.id.recyclerViewForTitle)

layoutManager1.orientation = LinearLayoutManager.HORIZONTAL

recyclerView1.layoutManager = layoutManager1

val adapter1 = TitleMenuAdapter(titleMenu)

recyclerView1.adapter = adapter1

//新闻栏配置

initFlowNewsMenu()

var layoutManager=StaggeredGridLayoutManager(3,StaggeredGridLayoutManager.VERTICAL)

val recyclerView: RecyclerView = findViewById(R.id.recyclerViewForGame)

recyclerView.layoutManager = layoutManager

val adapter = flow\_newsAdapter(flowNews)

recyclerView.adapter = adapter

}

private fun initFlowNewsMenu() {

repeat(5) {

flowNews.add(

flow\_news(

R.drawable.gamenews1,

"《黑神话：悟空》游戏早期概念图 猴子和白龙亮相\n《黑神话：悟空》游戏早期概念图 猴子和白龙亮相\n《黑神话：悟空》游戏早期概念图 猴子和白龙亮相",

"www.3dmgame.com",

"2020-10-22 11:35:53",

"https://www.3dmgame.com/news/202010/3800200.html",

"来不排除会有DLC和非数值向的内购。"

)

)

flowNews.add(

flow\_news(

R.drawable.gamenews2,

"《战争机器5》XSX输入延迟大幅降低 最高达57%\n《战争机器5》XSX输入延迟大幅降低 最高达57%",

"baijiahao.baidu.com",

"2020-10-22 08:29:47",

"https://baijiahao.baidu.com/s?id=1681212969147287593&wfr=spider&for=pc",

""

)

)

flowNews.add(

flow\_news(

R.drawable.gamenews3,

"《赛博朋克2077》角色使用各种语言都能实现口型同步",

"www.3dmgame.com",

"2020-10-21 00:02:28",

"https://www.3dmgame.com/news/202010/3800061.html",

""

)

)

}

}

private fun initTitleMenu() {

titleMenu.add(TitleMenu("Business", R.drawable.business))

titleMenu.add(TitleMenu("Entertainment", R.drawable.entertainment))

titleMenu.add(TitleMenu("Health", R.drawable.health))

titleMenu.add(TitleMenu("Science", R.drawable.science))

titleMenu.add(TitleMenu("Sport", R.drawable.sport))

titleMenu.add(TitleMenu("Game", R.drawable.game))

}

}

val recyclerView: RecyclerView = findViewById(R.id.recyclerViewForGame)

recyclerView.layoutManager = layoutManager

val adapter = flow\_newsAdapter(flowNews)

recyclerView.adapter = adapter

}

private fun initFlowNewsMenu() {

repeat(5) {

flowNews.add(

flow\_news(

R.drawable.gamenews1,

"《黑神话：悟空》游戏早期概念图 猴子和白龙亮相\n《黑神话：悟空》游戏早期概念图 猴子和白龙亮相\n《黑神话：悟空》游戏早期概念图 猴子和白龙亮相",

"www.3dmgame.com",

"2020-10-22 11:35:53",

"https://www.3dmgame.com/news/202010/3800200.html",

"来不排除会有DLC和非数值向的内购。"

)

)

flowNews.add(...)

flowNews.add(...)

}

}

private fun initTitleMenu() {

titleMenu.add(TitleMenu("Business", R.drawable.business))

titleMenu.add(TitleMenu("Entertainment", R.drawable.entertainment))

titleMenu.add(TitleMenu("Health", R.drawable.health))

titleMenu.add(TitleMenu("Science", R.drawable.science))

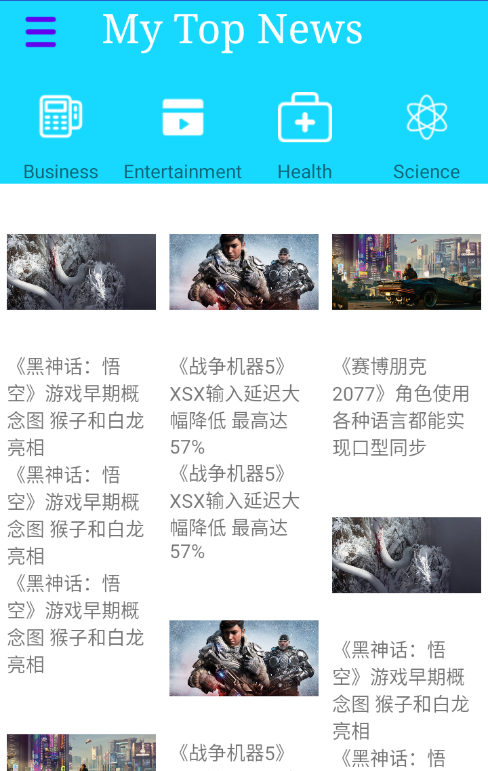
titleMenu.add(TitleMenu("Sport", R.drawable.sport))

titleMenu.add(TitleMenu("Game", R.drawable.game))

}

}

最终效果图：

**1.5 动态效果设计**

·通过点击标题栏的图标，实现不同新闻界面之间的切换。

·通过点击新闻栏，实现不同方式的新闻阅览。

**1.5.1 新闻界面切换**

·在标题栏适配器TitleMenuAdapter.kt中进行修改。在onCreateViewHolder（）方法中注册点击事件，为最外层布局注册点击事件，itemView表示的就是最外层布局。在两个点击事件中先获取用户点击的position，然后通过position拿到相应的实例，在使用intent完成activity之间的跳转。

·因为每个界面都设置了标题栏，所以多次点击会导致activity多次进入堆栈。解决方法是设置新闻界面的launchMode为singleTask。然后再点击前判断父活动是否为同一界面或是主菜单，如果是，则关闭上一个活动。

override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {

...

val viewHolder = ViewHolder(view)

viewHolder.titleMenuImage.setOnClickListener() {

val position: Int = viewHolder.layoutPosition

when(position){

0->{

val intent = Intent(parent.context, Business::class.java)

parent.context.startActivity(intent)

if (parent.context !is MainActivity && parent.context !is Business){

(parent.context as Activity).finish()

}

}

1->{

val intent = Intent(parent.context, Entertainment::class.java)

parent.context.startActivity(intent)

if (parent.context !is MainActivity && parent.context !is Entertainment){

(parent.context as Activity).finish()

}

}

2->{

val intent = Intent(parent.context, Health::class.java)

parent.context.startActivity(intent)

if (parent.context !is MainActivity && parent.context !is Health){

(parent.context as Activity).finish()

}

}

3->{

val intent = Intent(parent.context, Science::class.java)

parent.context.startActivity(intent)

if (parent.context !is MainActivity && parent.context !is Science){

(parent.context as Activity).finish()

}

}

4->{

val intent = Intent(parent.context, Sport::class.java)

parent.context.startActivity(intent)

if (parent.context !is MainActivity && parent.context !is Sport){

(parent.context as Activity).finish()

}

}

5->{

val intent = Intent(parent.context, Game::class.java)

parent.context.startActivity(intent)

if (parent.context !is MainActivity && parent.context !is Game){

(parent.context as Activity).finish()

}

}

}

}

}

}

1->{

val intent = Intent(parent.context, Entertainment::class.java)

parent.context.startActivity(intent)

if(parent.context!isMainActivity&&parent.context!is Entertainment){

(parent.context as Activity).finish()

}

}

2->{...}

3->{...}

4->{...}

5->{...}

}

}

}

**1.5.2 新闻阅览界面切换**

·采取三种新闻阅览方式，第一种是通过本地下载的标题和文字来进行展示，第二种方式是通过手机自带的浏览器进行展示，第三种是通过WebView控件，在APP内部进行展示。

·默认每个新闻界面第一条新闻通过本地下载进行展示，第二条通过机自带的浏览器进行展示，第三条通过WebView控件，在APP内部进行展示。剩余新闻均通过第三种方式进行展示。

·首先在标题栏适配器vertical\_newsAdapter.kt和flow\_newsAdapter.kt中进行修改（两处代码基本一致，只需要注意修改具体对应变量，以vertical\_newsAdapter.kt为例），过程与先前介绍方式大致相同，此处先将具体实现代码空着，后面讲提到具体实现方法。

override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {

val view = LayoutInflater.from(parent.context)

.inflate(R.layout.vertical\_news\_item, parent, false)

val viewHolder = ViewHolder(view)

viewHolder.itemView.setOnClickListener() {

val position: Int = viewHolder.layoutPosition

val verticalNews = vertical\_newsList[position]

when (position) {

//本地文本查看

0 -> {...}

//调用系统浏览器

1 -> {...}

//使用WebView

2 -> {...}

else->{...}

}

}

return viewHolder

}

·接下载介绍三种打开方式的具体实现

①本地下载方式

·创建新活动，New->Activity->Empty Activity,创建LocalNews.kt与activity\_local\_news.xml文件，首先打开activity\_local\_news.xml，编写布局代码，实现本地显示界面设计。

·使用ScrollView控件，方式文章因为篇幅过长而导致不能完全显示。

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

<LinearLayout 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"

android:orientation="vertical"

tools:context=".LocalNews">

<include layout="@layout/title"/>

<ScrollView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical">

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="center"

android:textSize="30sp"

android:fontFamily="sans-serif-black"

android:id="@+id/LocalTitle"/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:textSize="12sp"

android:id="@+id/LocalContent"/>

</LinearLayout>

</ScrollView>

</LinearLayout>

·在vertical\_newsAdapter.kt适配器中添加具体实现代码，使用intent完成activity之间的跳转，并通过putExtra（）方法传递数据。

.......

when (position) {

//本地文本查看

0 -> {

val intent = Intent(parent.context, LocalNews::class.java)

intent.putExtra("Title", verticalNews.newsTitle)

intent.putExtra("Content", verticalNews.content)

parent.context.startActivity(intent)

}

......

}

......

·修改LocalNews.kt代码，获取intent中传输的数据，并修改界面TextView的内容。

package com.example.experiment2

import androidx.appcompat.app.AppCompatActivity

import android.os.Bundle

import kotlinx.android.synthetic.main.activity\_local\_news.\*

class LocalNews : AppCompatActivity() {

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_local\_news)

supportActionBar?.hide()

val Title: String? =intent.getStringExtra("Title")

val Content: String?=intent.getStringExtra("Content")

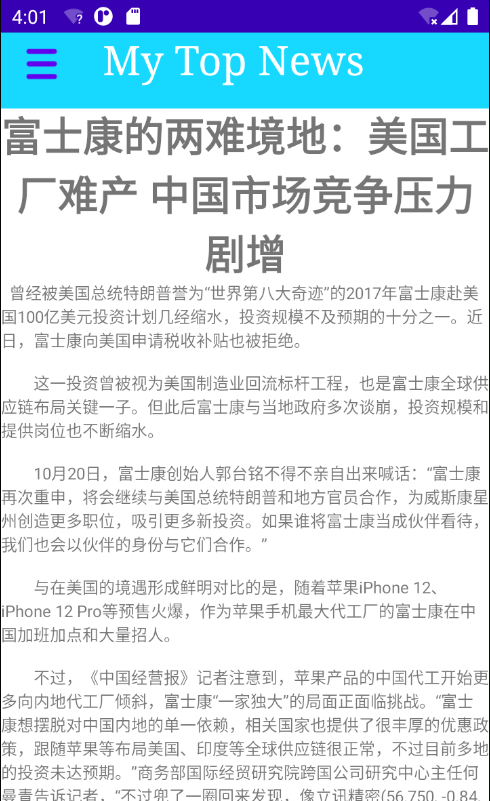
LocalTitle.text=Title

LocalContent.text=Content

}

}

最终效果图：



②自带浏览器方式

·在vertical\_newsAdapter.kt适配器中添加具体实现代码，通过intent跳转到手机自带浏览器进行阅览。

1 -> {

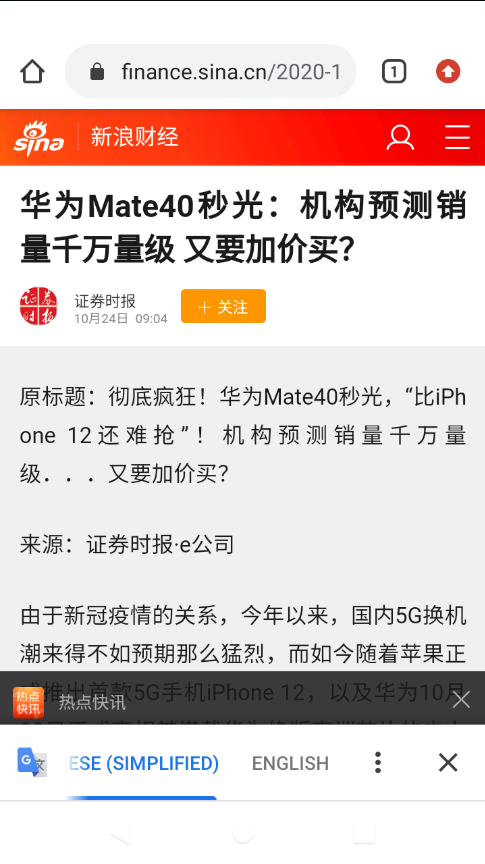
val intent = Intent(Intent.ACTION\_VIEW)

intent.data = Uri.parse(verticalNews.URL)

parent.context.startActivity(intent)

}

最终效果图：



③WebView控件方式

**·**创建新活动，New->Activity->Empty Activity,创建WebView.kt与activity\_web\_view.xml文件，首先打开activity\_web\_view.xml，编写布局代码。

·插入WebView控件，在应用程序里嵌入一个浏览器

<?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">

<include layout="@layout/title"/>

<WebView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@+id/webView"/>

</LinearLayout>

·在vertical\_newsAdapter.kt适配器中添加具体实现代码，使用intent完成activity之间的跳转，并通过putExtra（）方法传递URL数据。

2 -> {

val intent = Intent(parent.context, WebView::class.java)

intent.putExtra("URL", flowNews.URL)

parent.context.startActivity(intent)

}

·修改WebView.kt。通过WebView的getSettings（）方法可以设置一些浏览器的属性，这里并没有设置过多的属性，只是调用了色图JavaScript Enable（）方法，让WebView支持JavaScript脚本。接下来，调用setWebViewClient（）方法，并传入一个WebViewClient的实例，作用是当需要从一个网页跳转到另一个网页时，我们希望目标网页任然在当前WebView中显示，而不是打开系统浏览器。最后一步调用loadURL（）方法，并将网址传入，展示相应网页的内容。

package com.example.experiment2

import android.annotation.SuppressLint

import androidx.appcompat.app.AppCompatActivity

import android.os.Bundle

import android.webkit.WebViewClient

import kotlinx.android.synthetic.main.activity\_web\_view.\*

class WebView : AppCompatActivity() {

@SuppressLint("SetJavaScriptEnabled")

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_web\_view)

supportActionBar?.hide()

val URL: String? =intent.getStringExtra("URL")

setContentView(R.layout.activity\_web\_view)

webView.settings.javaScriptEnabled=true

webView.webViewClient= WebViewClient()

if (URL != null) {

webView.loadUrl(URL)

}

}

}

·需要注意，由于程序使用到了网络功能，而访问网络是需要声明权限的，因此还得修改AndroidManifest.xml文件，加入权限声明。

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

最终效果图：



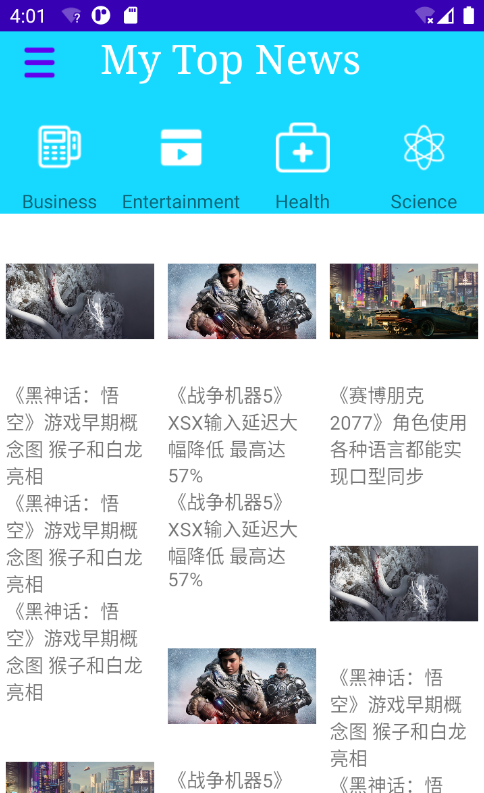
**2. 请详细说明“我的头条”APP的功能、出现的关键问题及解决方案**

**2.1 功能介绍**

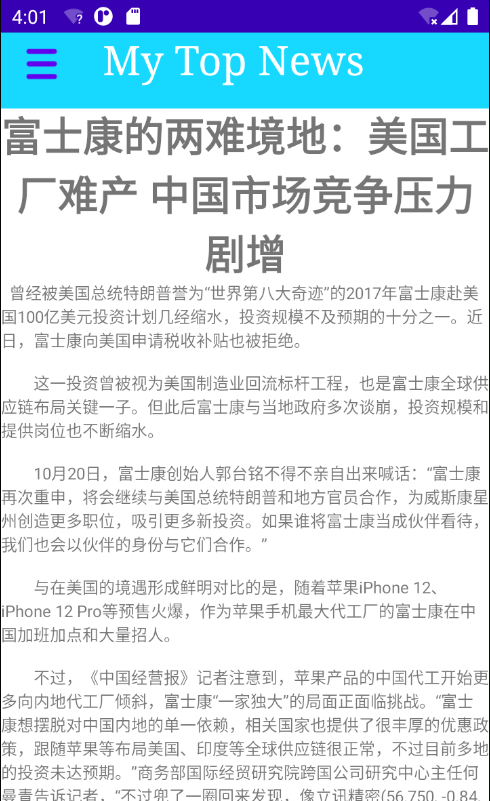
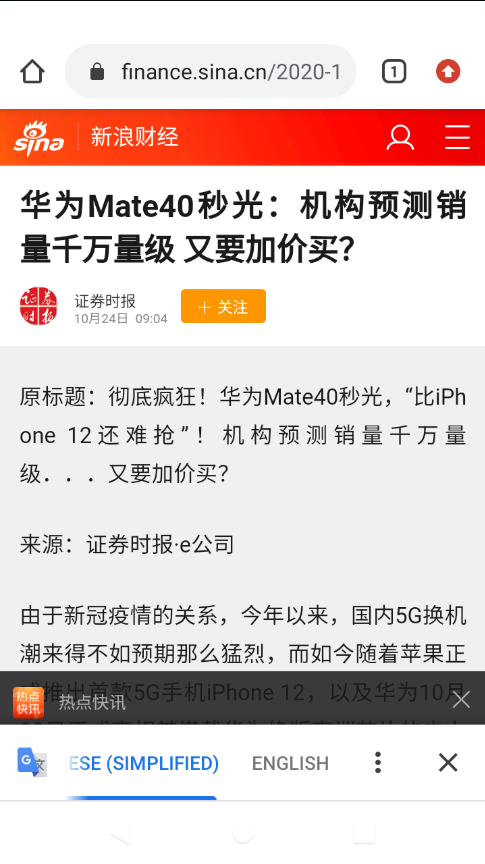
主界面：



新闻界面：（以Business与Game为例，展示垂直布局与瀑布布局）

新闻阅览界面：（第一条新闻通过本地下载进行展示，第二条通过机自带的浏览器进行展示，第三条通过WebView控件，在APP内部进行展示。）

**2.2 出现的关键问题及解决方案**

1. 问题：你的主机中的软件种植了一个已建立的连接。

解决方法：网络连接错误，重启电脑或网络。

1. 问题：AS同步慢

解决方法：



1. 问题：gradle文件更新下载，同不满

解决方法：查看一下在gradle的版本，修改gradle-wrapper.property中的版本

1. 问题：RecyclerView只出现一个图标

解决方法：item.xm中LinearLayout包含在了一个大的Layout中，应修改为只包含一个线性布局。

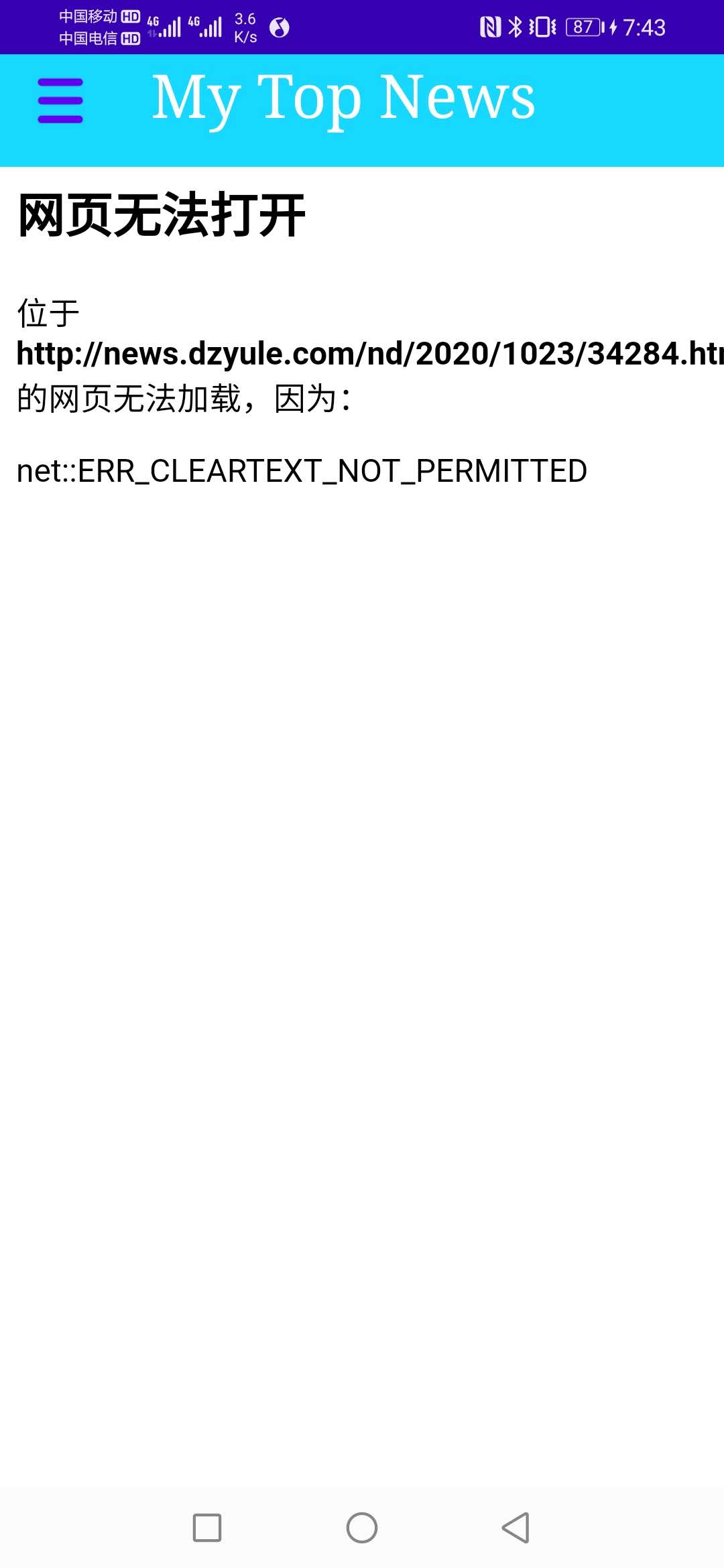
1. 问题：kotlin无法使用语法糖，省去findviewbyid等

解决方法：缺少kotlin依赖配置。gradle.build中添加语句：apply plugin ”kotlin-android-extensions”.

1. 问题：因为每个界面都设置了标题栏，所以多次点击会导致activity多次进入堆栈。

解决方法：设置新闻界面的launchMode为singleTask。然后再点击前判断父活动是否为 同一界面或是主菜单，如果是，则关闭上一个活动。

1. 问题：WebView无法打开网页。



解决方法：从Android 9.0（API级别28）开始，默认情况下禁用明文支持。因此http 的url均无法在webview中加载。在manifest 中application节点添加： android:usesCleartextTraffic="true"

1. 问题：文章篇幅过长，不能完全显示。

解决方法：使用ScrollView控件。

三、实验总结

本次实验结合了《第一行代码 Android》第二版第2章（活动）、第3章（UI开发）与第9章（网络技术）的各个知识点。大部分代码都是在书本上demo的基础上进行修改，完成了实验的基本需求。遇到的问题在先前已经进行了说明，并提出了解决方案。

|  |
| --- |
| 指导教师批阅意见：  成绩评定：  指导教师签字：解为成  年 月 日 |
| 备注： |

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。