МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования  
 «Кемеровский государственный университет»**

**Институт фундаментальных наук**

**Кафедра ЮНЕСКО по ИВТ**

**ЛАБОРАТОРНАЯ РАБОТА №10**

**ПО ДИСЦИПЛИНЕ “РАЗРАБОТКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ”**

студента 2 курса

**Крючкова Андрея Олеговича**

Направление 09.03.03 – Прикладная информатика в экономике

Преподаватель:

доцент

С.Ю. Завозкин

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Работа защищена:

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_202\_г.

с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_

Кемерово 202\_

СОДЕРЖАНИЕ

[1. Постановка задач 2](#_Toc87212842)

[2. Реализация задач 3](#_Toc87212843)

[Заключение 9](#_Toc87212844)

# 1. Постановка задач

1. Реализовать приложения, использующие Fragment для отображения данных на Activity, а также логирующие сетевое взаимодействие посредством класса Logging Interceptor .

# 2. Реализация задач

* 1. **Задание**

1)

<?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:id="@+id/main\_activity"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 tools:context=".MainActivity">  
  
 <androidx.appcompat.widget.Toolbar  
 android:id="@+id/toolbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="?attr/colorPrimary"  
 android:minHeight="?attr/actionBarSize"  
 android:theme="?attr/actionBarTheme">  
  
 <RelativeLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:padding="8dp">  
  
 <EditText  
 android:id="@+id/et\_search"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:inputType="textPersonName"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 </RelativeLayout>  
 </androidx.appcompat.widget.Toolbar>  
  
 <LinearLayout  
 android:id="@+id/fragment\_content"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical">  
  
 <androidx.fragment.app.FragmentContainerView  
 android:id="@+id/fragmentContainerView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content" />  
  
 </LinearLayout>  
  
  
</LinearLayout>

Код Activity\_main.xml

2)

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:id="@+id/rview\_item"  
 android:layout\_width="match\_parent"  
 android:layout\_height="99dp"  
 android:orientation="vertical">  
  
 <TextView  
 android:id="@+id/textName"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textPhone"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="TextView"  
 android:autoLink="phone"  
 android:background="@null"  
 android:linksClickable="true"  
 android:textAllCaps="true"  
 android:textIsSelectable="false"/>  
  
 <TextView  
 android:id="@+id/textType"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="TextView"  
  
 />  
</LinearLayout>

Код rview\_item.xml

3)

<?xml version="1.0" encoding="utf-8"?>  
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".Contact\_fragment">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical">  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/rView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 </androidx.recyclerview.widget.RecyclerView>  
 </LinearLayout>  
  
</FrameLayout>

contact\_fragment.xml

4) Перенесём логику работы recyclerView во фрагмент.

package com.example.mydialer  
  
import android.content.Context  
import android.os.Bundle  
import android.text.Editable  
import android.text.TextWatcher  
import androidx.fragment.app.Fragment  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.EditText  
import androidx.recyclerview.widget.LinearLayoutManager  
import androidx.recyclerview.widget.RecyclerView  
import com.google.gson.Gson  
import timber.log.Timber  
import java.io.IOException  
import java.io.InputStream  
  
  
class Contact\_fragment : Fragment() {  
  
 override fun onCreateView(  
 inflater: LayoutInflater, container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View? {  
 // Inflate the layout for this fragment  
 return inflater.inflate(R.layout.*contact\_fragment*, container, false)  
 }  
  
 @Override  
 override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  
 super.onViewCreated(view, savedInstanceState)  
  
 var adapter = Adapter()  
  
 var json: String? = null  
 val txt: EditText = requireActivity().findViewById(R.id.*et\_search*)  
 val recyclerView: RecyclerView = requireActivity().findViewById(R.id.*rView*)  
 Timber.plant(Timber.DebugTree())  
  
 try {  
 val inp: InputStream = requireActivity().*assets*.open("phones.json")  
 json = inp.*bufferedReader*().*use* **{ it**.*readText*() **}**.*replace*("\r\n", "")  
  
 val phones = Gson().fromJson(json, Array<Contact>::class.*java*).*toList*()  
  
 recyclerView.*layoutManager* = LinearLayoutManager(*context*)  
 recyclerView.*adapter* = adapter  
 adapter.submitList(phones)  
  
 txt.addTextChangedListener(object : TextWatcher {  
  
 override fun afterTextChanged(s: Editable) {  
  
 val sharedPreferences = requireActivity().getSharedPreferences("sharedPrefs", Context.*MODE\_PRIVATE*)  
 val editor = sharedPreferences.edit()  
 editor.*apply***{** putString("STRING\_KEY", txt.*text*.toString())  
 **}**.apply()  
  
 }  
  
 override fun beforeTextChanged(  
 s: CharSequence, start: Int,  
 count: Int, after: Int  
 ) {  
  
 }  
  
 override fun onTextChanged(  
 s: CharSequence, start: Int,  
 before: Int, count: Int  
 ) {  
 Thread **{** if (txt.*text*.toString() != "") {  
 val newArr = ArrayList<Contact>()  
 for (i in phones.*indices*) {  
 if (txt.*text*.toString() in phones[i].name || txt.*text*.toString() in phones[i].phone || txt.*text*.toString() in phones[i].type) {  
 newArr.add(phones[i])  
 }  
 }  
 requireActivity().runOnUiThread()**{** adapter.submitList(newArr)  
 adapter.notifyDataSetChanged()  
 **}** } else {  
 requireActivity().runOnUiThread()**{** adapter.submitList(phones)  
 adapter.notifyDataSetChanged()  
 **}** }  
  
 **}**.start()  
 }  
 })  
 } catch (e: IOException) {  
 Timber.e(e)  
 }  
  
  
 }  
}

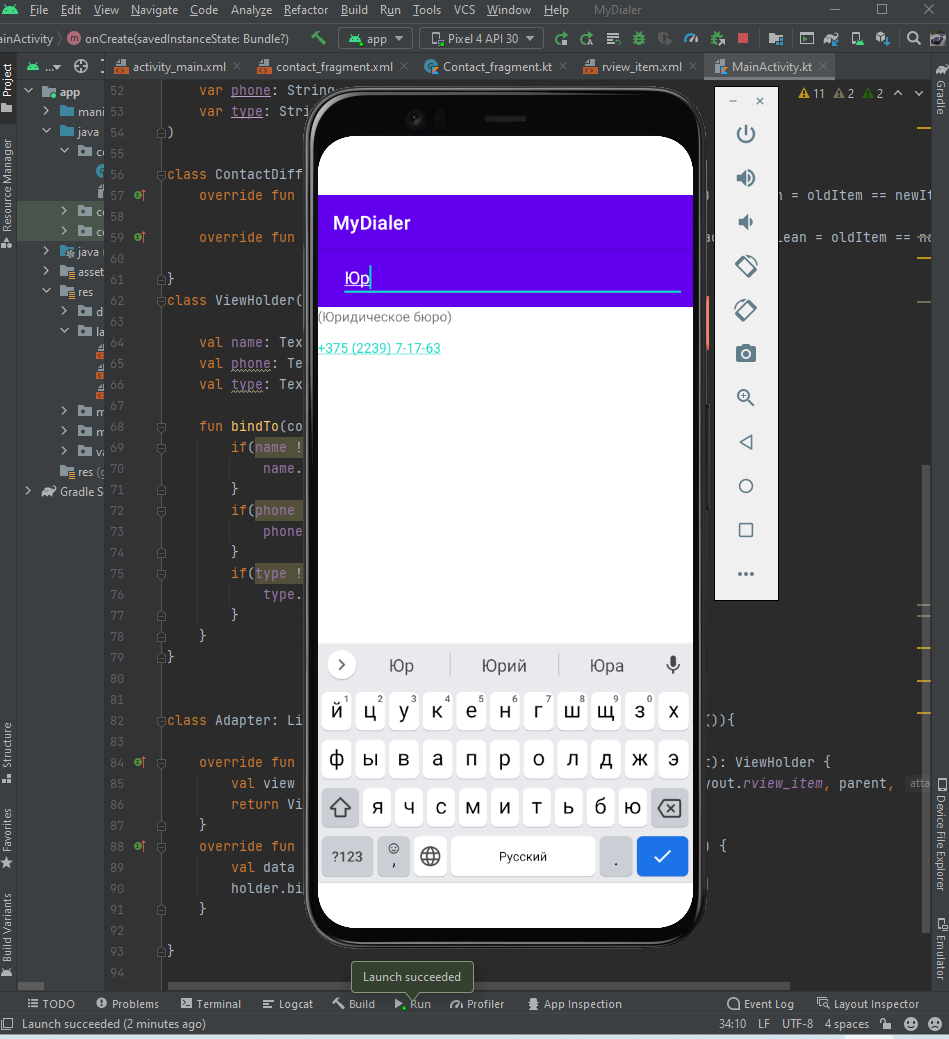
Класс фрагмента Contact\_fragment.kt

5)

class ContactItemDiffCallback : DiffUtil.ItemCallback<ListItem>(){  
 override fun areItemsTheSame(oldItem: ListItem, newItem: ListItem) = oldItem == newItem  
  
 override fun areContentsTheSame(oldItem: ListItem, newItem: ListItem) = oldItem == newItem  
  
}  
  
class ViewHolderCold(view: View) : RecyclerView.ViewHolder(view) {  
 val datetime: TextView = view.findViewById(R.id.*datetime\_text*)  
 val temperature: TextView = view.findViewById(R.id.*temperature\_text*)  
  
 fun bindTo(weather: ListItem, position: Int){  
 if (temperature != null) {temperature.*text* = weather.main.temp.toString()}  
 if (datetime != null) {datetime.*text* = weather.dt\_txt}  
  
 }  
}  
  
class ViewHolderHot(view: View) : RecyclerView.ViewHolder(view) {  
 val datetime: TextView = view.findViewById(R.id.*datetime\_text*)  
 val temperature: TextView = view.findViewById(R.id.*temperature\_text*)  
  
  
  
 fun bindTo(weather: ListItem){  
 if (temperature != null) {temperature.*text* = weather.main.temp.toString()}  
 if (datetime != null) {datetime.*text* = weather.dt\_txt}  
  
 }  
}  
  
class Adapter : ListAdapter<ListItem, RecyclerView.ViewHolder>(ContactItemDiffCallback()){  
  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): RecyclerView.ViewHolder {  
 if(viewType == 0){ val view = LayoutInflater.from(parent.*context*).inflate(R.layout.*rview\_item*, parent, false)  
 view.setBackgroundColor(Color.rgb(208, 65, 65))  
 return ViewHolderHot(view)}  
 else{  
 val view = LayoutInflater.from(parent.*context*).inflate(R.layout.*rview\_item2*, parent, false)  
 view.setBackgroundColor(Color.rgb(26, 103, 212))  
 return ViewHolderCold(view)  
 }  
 }  
 override fun onBindViewHolder(holder: RecyclerView.ViewHolder, position: Int) {  
 val data = *currentList*[position]  
 if(data.main.temp>0) {  
 val viewholderhot: ViewHolderHot = holder as ViewHolderHot  
 viewholderhot.bindTo(data)  
 }else{  
 val viewholdercold: ViewHolderCold = holder as ViewHolderCold  
 viewholdercold.bindTo(data,position)  
 }  
 }  
 override fun getItemViewType(position: Int): Int {  
 return if (*currentList*[position].main.temp>0) {  
 0  
 } else {  
 1  
 }  
 }  
}

Adaper.kt

Итоговое приложение



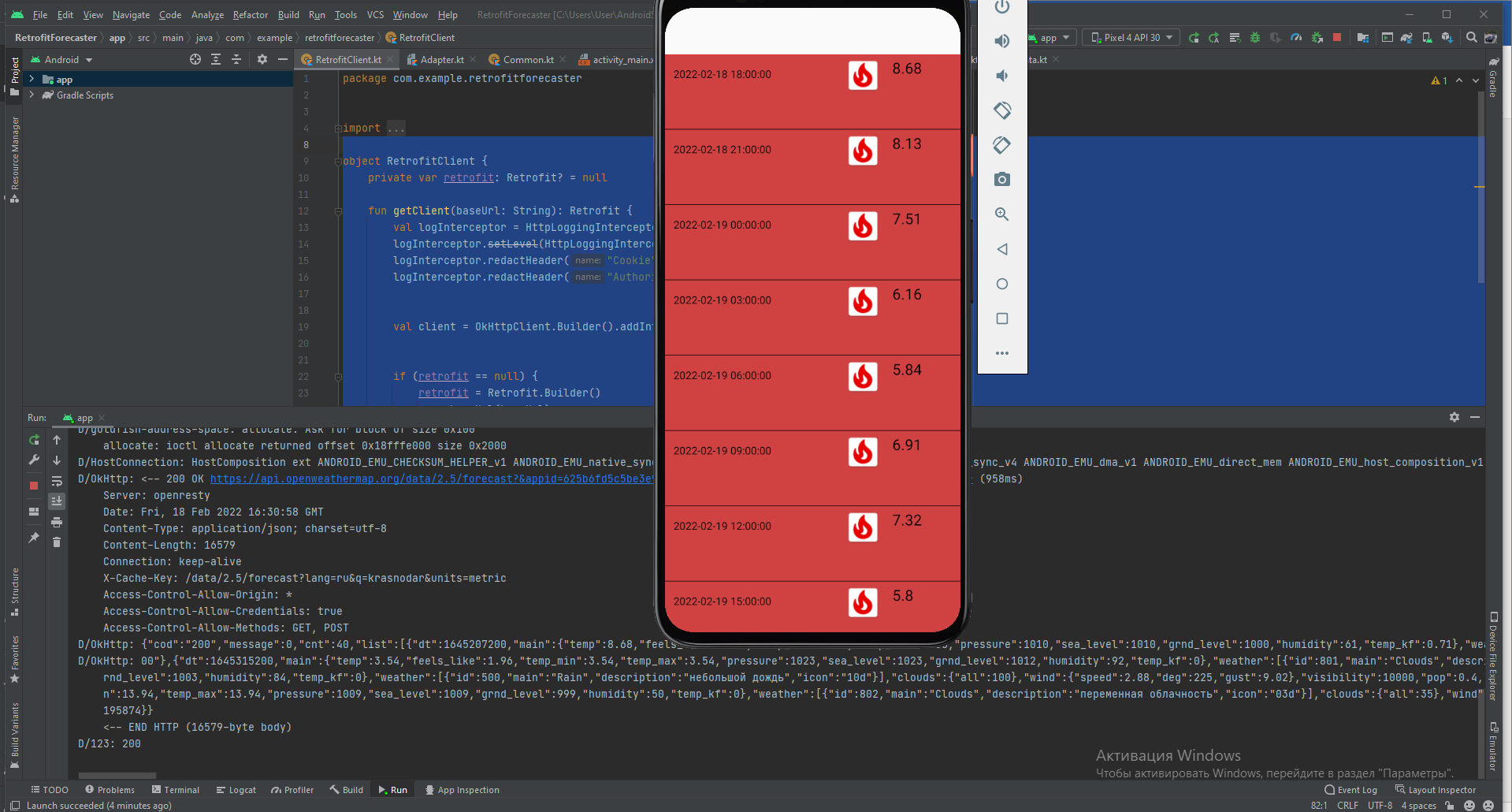
* 1. **Задание**

1. Добавим перехватчик HttpLoggingInterceptor() в RetrofitClient, заблокировав логирование Cookie и Authorization, и передадим логирующий перехватчик в клиент Retrofit’а.

object RetrofitClient {  
 private var retrofit: Retrofit? = null  
  
 fun getClient(baseUrl: String): Retrofit {  
 val logInterceptor = HttpLoggingInterceptor()  
 logInterceptor.setLevel(HttpLoggingInterceptor.Level.*BODY*)  
 logInterceptor.redactHeader("Cookie")  
 logInterceptor.redactHeader("Authorization")  
  
  
 val client = OkHttpClient.Builder().addInterceptor(logInterceptor).build()  
  
  
 if (retrofit == null) {  
 retrofit = Retrofit.Builder()  
 .baseUrl(baseUrl)  
 .client(client)  
 .addConverterFactory(GsonConverterFactory.create())  
 .build()  
 }  
 return retrofit!!  
 }  
}

Код RetrofitClient.kt

2) Итоговое приложение и логирование



# Заключение

Я ознакомился с фрагментами. Научился их создавать и отображать внутри активности. А также логировать сетевое взаимодействие.

**Литература**

1. Fragments  - <https://developer.android.com/guide/fragments>
2. Фрагменты. Введение во фрагменты - <https://metanit.com/java/android/8.1.php>
3. Повторное создание Activity - <https://www.fandroid.info/povtornoe-sozdanie-deyatelnosti/>
4. Interceptors - <https://square.github.io/okhttp/interceptors/>
5. Retaining Objects Using a Fragment - <https://pspdfkit.com/blog/2019/retaining-objects-using-a-fragment/>
6. Лекция 1. Введение в архитектуру клиент-серверных андроид-приложений. Часть 2 - <https://www.fandroid.info/lektsiya-1-vvedenie-v-arhitekturu-klient-servernyh-android-prilozhenij-chast-2/#4>Android RecyclerView with multiple view type (multiple view holder) - <https://droidbyme.medium.com/android-recyclerview-with-multiple-view-type-multiple-view-holder-af798458763b>