**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное учреждение

высшего образования «Уральский федеральный университет

имени первого Президента России Б.Н. Ельцина»

Институт радиоэлектроники и информационных технологий - РТФ

Центр ускоренного обучения

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ 24**

по дисциплине «Разработка мобильных приложений»

**Тема: Unit 3 Работа в фоновом режиме “Расписание заданий”**

Студент группы РИЗ-300028у: А.С. Севостьянов

Преподаватель: С.И.Тимошенко,

доц., к.т.н.

**Екатеринбург 2022**

содержание

[1 Постановка задачи 3](#_Toc118041527)

[2 Результаты проделанной работы 4](#_Toc118041528)

[3 Анализ полученных результатов 5](#_Toc118041529)

## Постановка задачи

Получить ввод данных пользователя для настройки ограничений (например, ожидания зарядки устройства) на JobService. Запланировать выполнение задания с помощью JobScheduler.

.

## Результаты проделанной работы

Программный код, файл MainActivity.

package com.android.example.notificationscheduler;  
  
import android.app.job.JobInfo;  
import android.app.job.JobScheduler;  
import android.content.ComponentName;  
import android.os.Bundle;  
import android.support.v7.app.AppCompatActivity;  
import android.view.View;  
import android.widget.RadioGroup;  
import android.widget.SeekBar;  
import android.widget.Switch;  
import android.widget.TextView;  
import android.widget.Toast;  
  
public class MainActivity extends AppCompatActivity {  
  
 private static final int *JOB\_ID* = 0;  
 private JobScheduler mScheduler;  
  
 // Переключатели для настройки параметров задания.  
 private Switch mDeviceIdleSwitch;  
 private Switch mDeviceChargingSwitch;  
  
 // Переопределить панель поиска крайнего срока.  
 private SeekBar mSeekBar;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 mDeviceIdleSwitch = findViewById(R.id.*idleSwitch*);  
 mDeviceChargingSwitch = findViewById(R.id.*chargingSwitch*);  
 mSeekBar = findViewById(R.id.*seekBar*);  
  
 final TextView seekBarProgress = findViewById(R.id.*seekBarProgress*);  
  
 mScheduler = (JobScheduler) getSystemService(*JOB\_SCHEDULER\_SERVICE*);  
  
 // Обновляет текстовое представление значением из панели поиска.  
 mSeekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {  
 @Override  
 public void onProgressChanged(SeekBar seekBar, int i, boolean b) {  
 if (i > 0) {  
 seekBarProgress.setText(getString(R.string.*seconds*, i));  
 } else {  
 seekBarProgress.setText(R.string.*not\_set*);  
 }  
 }  
  
 @Override  
 public void onStartTrackingTouch(SeekBar seekBar) {  
 }  
  
 @Override  
 public void onStopTrackingTouch(SeekBar seekBar) {  
 }  
 });  
  
 }  
  
 */\*\*  
 \* Метод onClick, который планирует задания на основе заданных параметров.  
 \*/* public void scheduleJob(View view) {  
 RadioGroup networkOptions = findViewById(R.id.*networkOptions*);  
  
 int selectedNetworkID = networkOptions.getCheckedRadioButtonId();  
  
 int selectedNetworkOption = JobInfo.*NETWORK\_TYPE\_NONE*;  
  
 int seekBarInteger = mSeekBar.getProgress();  
 boolean seekBarSet = seekBarInteger > 0;  
  
  
 switch (selectedNetworkID) {  
 case R.id.*noNetwork*:  
 selectedNetworkOption = JobInfo.*NETWORK\_TYPE\_NONE*;  
 break;  
 case R.id.*anyNetwork*:  
 selectedNetworkOption = JobInfo.*NETWORK\_TYPE\_ANY*;  
 break;  
 case R.id.*wifiNetwork*:  
 selectedNetworkOption = JobInfo.*NETWORK\_TYPE\_UNMETERED*;  
 break;  
 }  
  
 ComponentName serviceName = new ComponentName(getPackageName(),  
 NotificationJobService.class.getName());  
 JobInfo.Builder builder = new JobInfo.Builder(*JOB\_ID*, serviceName)  
 .setRequiredNetworkType(selectedNetworkOption)  
 .setRequiresDeviceIdle(mDeviceIdleSwitch.isChecked())  
 .setRequiresCharging(mDeviceChargingSwitch.isChecked());  
  
 if (seekBarSet) {  
 builder.setOverrideDeadline(seekBarInteger \* 1000);  
 }  
 boolean constraintSet = selectedNetworkOption  
 != JobInfo.*NETWORK\_TYPE\_NONE* || mDeviceChargingSwitch.isChecked()  
 || mDeviceIdleSwitch.isChecked()  
 || seekBarSet;  
  
 if (constraintSet) {  
 JobInfo myJobInfo = builder.build();  
 mScheduler.schedule(myJobInfo);  
 Toast.*makeText*(this, R.string.*job\_scheduled*, Toast.*LENGTH\_SHORT*)  
 .show();  
 } else {  
 Toast.*makeText*(this, R.string.*no\_constraint\_toast*,  
 Toast.*LENGTH\_SHORT*).show();  
 }  
 }  
  
 */\*\*  
 \* Метод onClick для отмены всех существующих заданий.  
 \*/* public void cancelJobs(View view) {  
  
 if (mScheduler != null) {  
 mScheduler.cancelAll();  
 mScheduler = null;  
 Toast.*makeText*(this, R.string.*jobs\_canceled*, Toast.*LENGTH\_SHORT*)  
 .show();  
 }  
 }  
}

Программный код, файл NotificationJobService.

package com.android.example.notificationscheduler;  
  
import android.app.NotificationChannel;  
import android.app.NotificationManager;  
import android.app.PendingIntent;  
import android.app.job.JobParameters;  
import android.app.job.JobService;  
import android.content.Intent;  
import android.graphics.Color;  
import android.support.v4.app.NotificationCompat;  
  
  
/\*  
 Служба, которую JobScheduler запускает после выполнения условий.  
 В этом случае он отправляет уведомление.  
 \*/  
public class NotificationJobService extends JobService {  
  
 // Идентификатор канала уведомления.  
 private static final String *PRIMARY\_CHANNEL\_ID* =  
 "primary\_notification\_channel";  
 // Менеджер уведомлений.  
 NotificationManager mNotifyManager;  
  
 */\*\*  
 Вызывается системой, как только она определяет, что пришло время выполнить задание.  
 \*/* @Override  
 public boolean onStartJob(JobParameters jobParameters) {  
  
 // Создайте канал уведомлений.  
 createNotificationChannel();  
  
 // Настройте намерение содержимого уведомления для запуска приложения, когда  
 // нажал.  
 PendingIntent contentPendingIntent = PendingIntent.*getActivity* (this, 0, new Intent(this, MainActivity.class),  
 PendingIntent.*FLAG\_UPDATE\_CURRENT*);  
  
 NotificationCompat.Builder builder = new NotificationCompat.Builder  
 (this, *PRIMARY\_CHANNEL\_ID*)  
 .setContentTitle(getString(R.string.*job\_service*))  
 .setContentText(getString(R.string.*job\_running*))  
 .setContentIntent(contentPendingIntent)  
 .setSmallIcon(R.drawable.*ic\_job\_running*)  
 .setPriority(NotificationCompat.*PRIORITY\_HIGH*)  
 .setDefaults(NotificationCompat.*DEFAULT\_ALL*)  
 .setAutoCancel(true);  
  
 mNotifyManager.notify(0, builder.build());  
 return false;  
 }  
  
 */\*\*  
 Вызывается системой, когда задание выполняется, но условия  
 больше не выполняются.  
 \*/* @Override  
 public boolean onStopJob(JobParameters jobParameters) {  
 return false;  
 }  
  
 */\*\*  
 \* Создает канал уведомлений для OREO и выше.  
 \*/* public void createNotificationChannel() {  
  
// Создайте объект диспетчера уведомлений.  
 mNotifyManager =  
 (NotificationManager) getSystemService(*NOTIFICATION\_SERVICE*);  
  
// Каналы уведомлений доступны только в OREO и выше.  
 // Итак, добавьте проверку версии SDK.  
 if (android.os.Build.VERSION.*SDK\_INT* >=  
 android.os.Build.VERSION\_CODES.*O*) {  
  
 // // Создайте канал уведомлений со всеми параметрами.  
 NotificationChannel notificationChannel = new NotificationChannel  
 (*PRIMARY\_CHANNEL\_ID*,  
 getString(R.string.*job\_service\_notification*),  
 NotificationManager.*IMPORTANCE\_HIGH*);  
  
 notificationChannel.enableLights(true);  
 notificationChannel.setLightColor(Color.*RED*);  
 notificationChannel.enableVibration(true);  
 notificationChannel.setDescription  
 (getString(R.string.*notification\_channel\_description*));  
  
 mNotifyManager.createNotificationChannel(notificationChannel);  
 }  
 }  
}

На рисунке 1 показано начальное состояние программы.

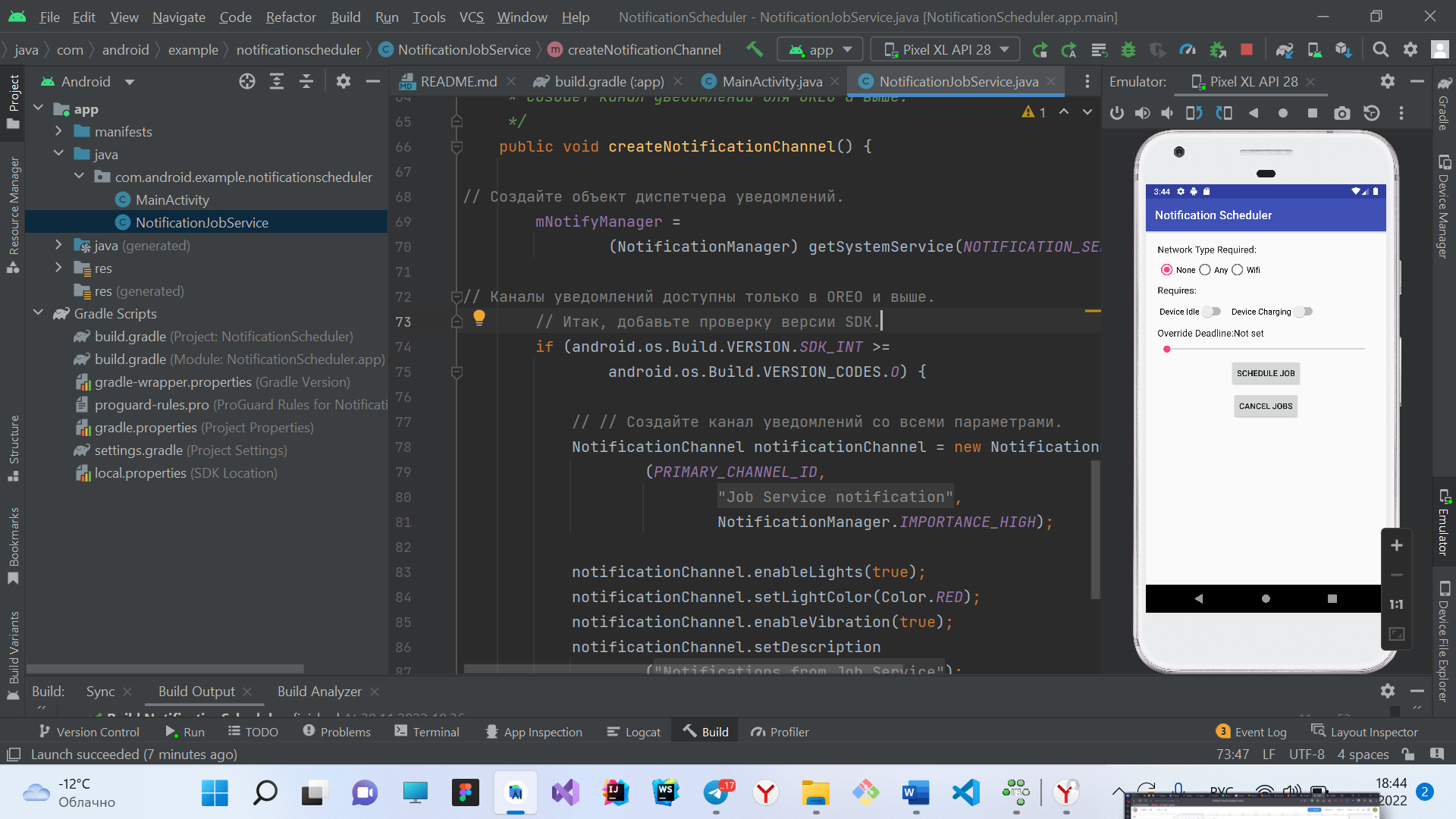


Рисунок 1 – Начальное состояние программы

На рисунке 2 показан результат работы программы.

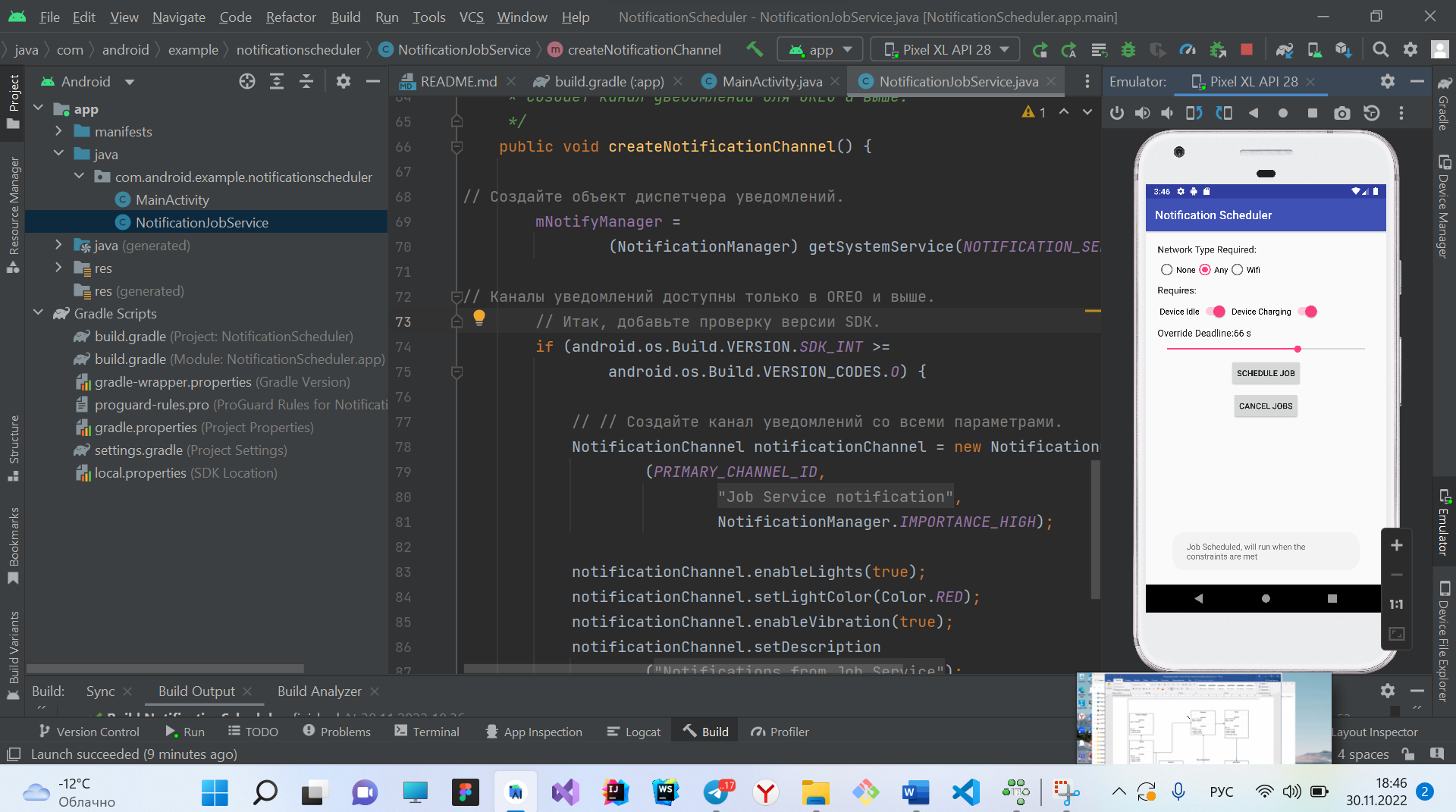


Рисунок 2 – Результат работы программы

## **Анализ полученных результатов**

В результате выполнения данной лабораторной работы мной был разработан проект, в котором я научился, реализовать a JobService. Создавать Jobinfo объект с определенными ограничениями. Планировать a JobService на основе Jobinfo объекта.

.