Разработка мобильных приложений

Практическая работа №4

Выполнил: Алеев А.В. БСБО-07-22

В ходе данной работы был создан проект «ru.mirea.AleevAV.Lesson4», в котором были созданы следующие модули «cryptoloader», «data_thread», «looper», «MusicPlayer», «serviceapp», «thread» и «workmanager» (см. рис .1).

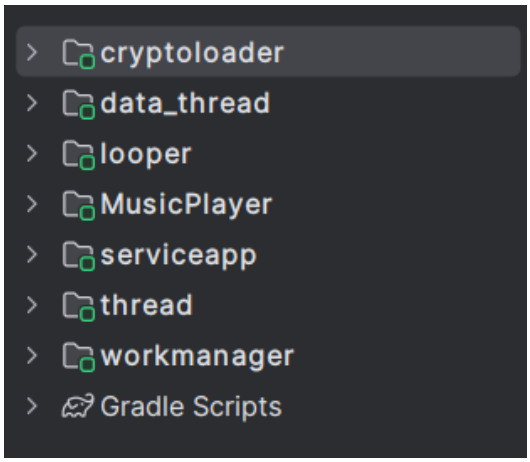

Рисунок 1. Модули проекта

В первом модуле «MusicPlayer» было создано две активности. В первой было выставлено текст, поле для ввода и кнопка, перенаправляющая на вторую активность (см. рис. 2 и лситинг 1). Все обращения были сделаны через «binding».

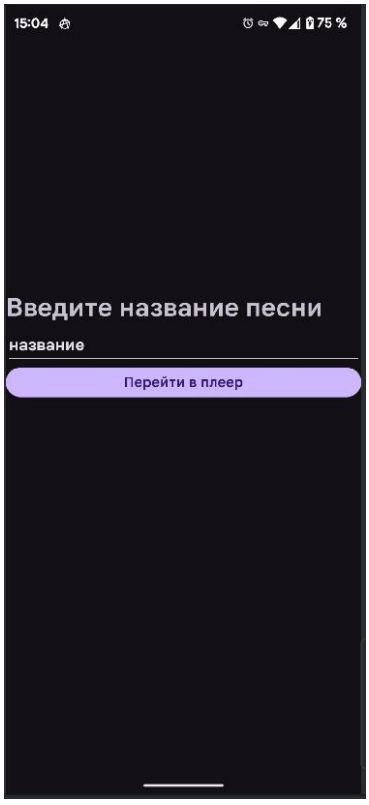

Рисунок 2. Первая активность

```
private ActivityMainBinding binding;

@Override
protected void onCreate(Bundle savedInstanceState) {
```

```
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable(this);
    setContentView(R.layout.activity_main);
    ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v,
insets) -> {
        Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
        v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
        return insets;
    });
    binding = ActivityMainBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());
    binding.textViewMirea.setText("Введите название песни");
    if (getIntent() != null){
        Intent name = getIntent();
        binding.editTextMirea.setText(name.getStringExtra("music_name"));
    }

    binding.editTextMirea.setHint("Название песни");


    binding.buttonMirea.setText("Перейти в плеер");
    binding.buttonMirea.setOnClickListener(new View.OnClickListener(){
        @Override
        public void onClick(View view){
            Intent intent = new Intent(view.getContext(),
MusicActivity.class);
            intent.putExtra("music_name",
binding.editTextMirea.getText().toString());
            Log.d(MainActivity.class.getSimpleName(), "onClickListener");
            startActivity(intent);
        }
    });
}
```

Литсинг 1. Код класса первой активности

Во второй активности был сделан экран музыкального плеера. В нём присутствуют текст, картинка, ползунок три кнопки плеера и одна кнопка, возвращающая к первой активности. В классе активности также все обращения к элементам пользовательского интерфейса были сделаны через «binding» (см. рис 3 и литсинг 2).

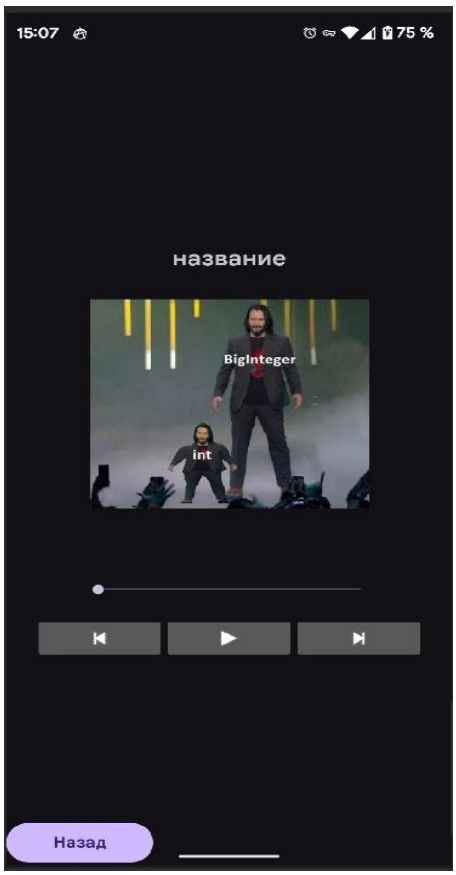

Рисунок 3. Экран плеера

```
private ActivityMusicBinding binding;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable(this);
    setContentView(R.layout.activity_music);
    ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v,
insets) -> {
        Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
        v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
        return insets;
    });

    binding = ActivityMusicBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());
    Intent info = getIntent();
    binding.textMusicName.setText(info.getStringExtra("music_name"));

    binding.buttonBack.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent(v.getContext(), MainActivity.class);
            intent.putExtra("music_name",
binding.textMusicName.getText().toString());
            startActivity(intent);
        }
    });
}
```

Листинг 2. Код класса второй активности

Также всем двум активностям были сделаны горизонтальные аналоги (см. рис 4).

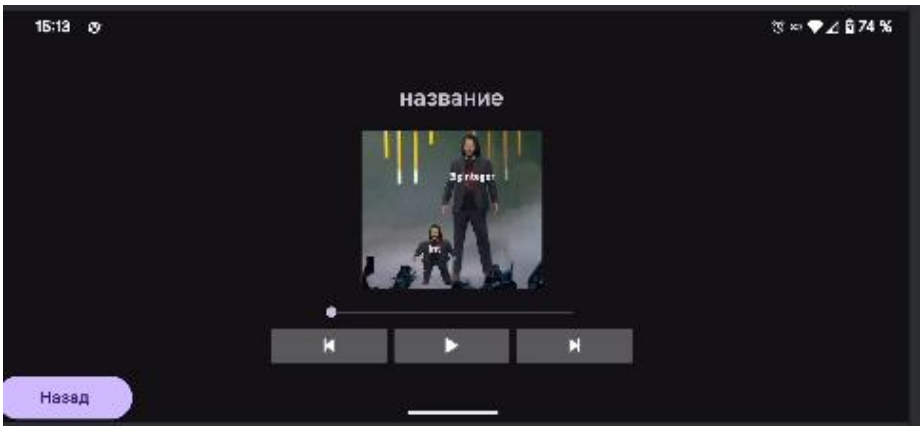

Рисунок 4. Пример горизонтального вида

Далее был создан второй модуль «thread». В нём при инициализации заменяется имя главного потока. Также есть возможность посчитать в фоновом потоке среднее количество пар в день за период одного месяца (см. рис 5 и листинг 3).

Рисунок 5. Пример выполнения подсчётов

```
public class MainActivity extends AppCompatActivity {
    private ActivityMainBinding binding;
    private Handler handler;

    private int counter;


    private int allLessons;

    private int allLessondays;
    private double result;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        binding = ActivityMainBinding.inflate(getLayoutInflater());


        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main),
(v, insets) -> {
            Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
            return insets;
        });
        setContentView(binding.getRoot());



        Thread mainthread = Thread.currentThread();
        binding.textViewThread.setText("Имя текущего потока: " +
mainthread.getName());

        mainthread.setName("Мой номер группы: БСБО-07022, мой номер по
списку: 1, мой любимый фильм: Шоу Трумана");
        binding.textViewThread.append("\n Новое имя потока: " +
mainthread.getName());
        Log.d("поток","Stack: " +
Arrays.toString(mainthread.getStackTrace()));

        binding.button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                new Thread(new Runnable() {
                    @Override
                    public void run() {

                        int numberthread = counter++;
                        Log.d("ПотокПроект",String.format("Запущен поток № %d
студентом группы № %s номер по списку № %d", numberthread, "БСБО-07-22", 1));
                        long endTime = System.currentTimeMillis() + 2 * 1000;
                        while(System.currentTimeMillis() < endTime){
                            synchronized (this){
                                try{
                                    allLessons =
Integer.valueOf(String.valueOf(binding.editTextLessons.getText()));
                                    allLessondays =
Integer.valueOf(String.valueOf(binding.editTextDays.getText()));
                                    wait(endTime -
```

```
System.currentTimeMillis());

                                    Log.d(MainActivity.class.getSimpleName(),
"Endtime: "+ endTime);
                                    result = (double) allLessons
/allLessondays;
                                    Log.d("ПотокПроект", ""+result);
                                    runOnUiThread(new Runnable() {
                                        @Override
                                        public void run() {

binding.textViewRes.setText("среднее кол-во пар за месяц: "+ result);
                                        }
                                    });

                                }catch (Exception e){
                                    throw new RuntimeException(e);
                                }
                            }
                        }
                    }

                }).start();

            }
        });

    }


}
```

Листинг 3. Запуск фоновых процессов

В модуле «data_thread» было определено в какой последовательности происходит запуск процессов. Спустя 2 секунды запускается 1-ый процесс, меняющий текст на runn1, затем спустя 1 секунду запускается 2-ой проце, после чего 3-й, так как при помощи метода «postDelayed» процесс «runn3» был поставлен в очередь с задержкой в 2 секунды (см. рис. 6 и листинг 5).

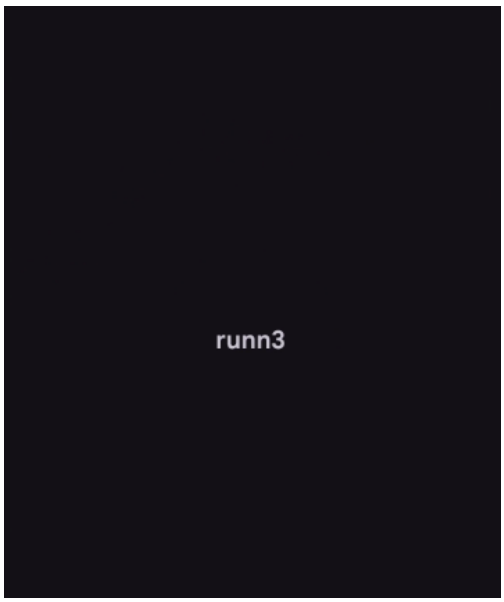

Рисунок 6. Пример работы

```
public class MainActivity extends AppCompatActivity {
    private ActivityMainBinding binding;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main),
(v, insets) -> {
            Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
            return insets;
        });

        binding = ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        final Runnable runn1 = new Runnable() {
            @Override
            public void run() {
                binding.TextView.setText("runn1");
            }
        };

        final Runnable runn2 = new Runnable() {
            @Override
            public void run() {
                binding.TextView.setText("runn2");
            }
        };

        final Runnable runn3 = new Runnable() {
            @Override
            public void run() {
                binding.TextView.setText("runn3");
            }
        };

        Thread t = new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    TimeUnit.SECONDS.sleep(2);
                    runOnUiThread(runn1);
                    TimeUnit.SECONDS.sleep(1);
                    binding.TextView.postDelayed(runn3, 2000);
                    binding.TextView.post(runn2);
                }catch (InterruptedException e){
                    e.printStackTrace();
                }
            }
        });
        t.start();
    }
}
```

Листинг 4. Код класса модуля «data_thread»

Далее был создан модуль «looper», в котором вводится возраст и работа. После нажатия на кнопку происходит задержка, равная возрасту, после

которой выводится в логах сообщение с возрастом и работой. Данная работа была выполнена с помощью «looper» (см. рис. 7 и листинг 5).

Рисунок 7. Пример вывода данных

```
public class MyLooper extends Thread{
    public Handler mHandler;
    private Handler mainHandler;
    public MyLooper(Handler mainThreadHandler){
        mainHandler = mainThreadHandler;
    }
    public void run(){
        Log.d("MyLooper","run");
        Looper.prepare();
        mHandler = new Handler(Looper.myLooper()){
            public void handleMessage(Message msg){
                Integer age = msg.getData().getInt("AGE");
                Log.d("MyLooper get message", String.valueOf(age));
                String Job = msg.getData().getString("JOB");
                Log.d("MyLooper get message", Job);

                Message message = new Message();
                Bundle bundle = new Bundle();
                try{
                    Thread.sleep(age * 1000);
                } catch (InterruptedException e){
                    e.printStackTrace();
                }


                bundle.putString("result",String.format("Возраст: %d |||
Работа: %s", age, Job));
                message.setData(bundle);

                mainHandler.sendMessage(message);

            }
        };
        Looper.loop();

    }
}
```

Листинг 5. Класс «MyLooper»

В следующем модуле «cryptoloader» было реализовано шифрование и дешифрование сообщения в фоне при помощи «loader». При нажатии на кнопку создаётся «loader», в который передаётся ключ и зашифрованное сообщение. Затем он дешифрует сообщение и появляется сообщение (см. рис 8 и листинг 6 и 7).

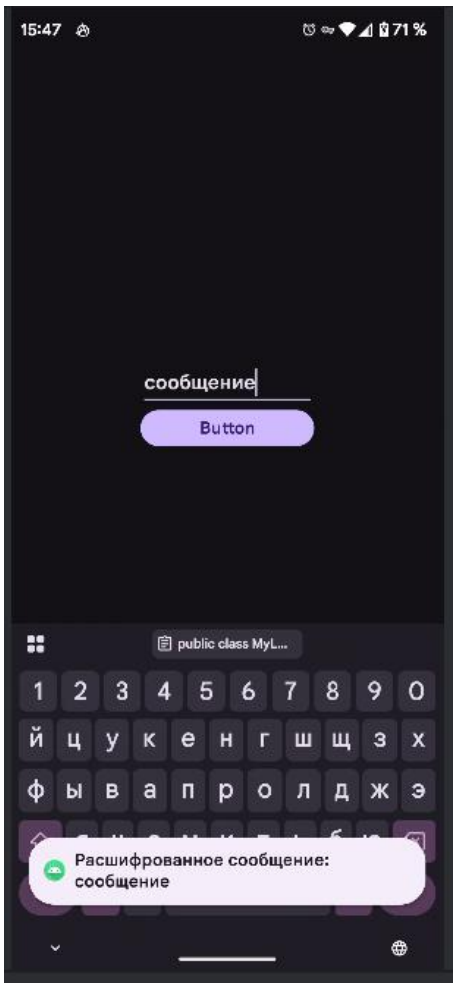

Рисунок 8. Пример расшифрования сообщения

```
public class MainActivity extends AppCompatActivity implements
LoaderManager.LoaderCallbacks<String> {
    private final int LoaderID = 1234;
    private ActivityMainBinding binding;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main),
(v, insets) -> {
            Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
            return insets;
        });
        binding = ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());
    }

    public void onClickEnc(View view){
        Bundle bundle = new Bundle();
        //bundle.putString(MyLoader.ARG_WORD,
binding.editTextText.getText().toString());
        //LoaderManager.getInstance(this).initLoader(LoaderID, bundle, this);
        SecretKey key = generateKey();
        bundle.putByteArray(MyLoader.ARG_WORD,
encryptMsg(binding.editTextText.getText().toString(), key));
```

```
        bundle.putByteArray("Key", key.getEncoded());
        LoaderManager.getInstance(this).restartLoader(LoaderID, bundle,
this);

    }


    @Override
    public void onLoaderReset(@NonNull Loader<String> loader){
        Log.d("Loader", "OnLoaderReset");
    }

    @NonNull
    @Override
    public Loader<String> onCreateLoader(int id, @Nullable Bundle bundle){
        if(id== LoaderID){
            Toast.makeText(this, "onCreateLoader: " + id,
Toast.LENGTH_SHORT).show();
            return new MyLoader(this, bundle);
        }
        throw new InvalidParameterException("Invalid loader id");

    }

    @Override
    public void onLoadFinished(@NonNull Loader<String> loader, String s) {
        if(loader.getId()==LoaderID){
            Log.d("Loader", "On loaderFinished: " + s);
            Toast.makeText(this, "Расшифрованное сообщение: " + s,
Toast.LENGTH_SHORT).show();
        }

    }

    public static SecretKey generateKey() {
        try{
            SecureRandom random = SecureRandom.getInstance("SHA1PRNG");
            random.setSeed("Metal Gear Solid".getBytes());
            KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
            keyGenerator.init(256, random);
            return keyGenerator.generateKey();

        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }
    public static byte[] encryptMsg(String message, SecretKey key) {
        Cipher cipher = null;
        try {
            cipher = Cipher.getInstance("AES");
            cipher.init(Cipher.ENCRYPT_MODE, key);
            return cipher.doFinal(message.getBytes());
        } catch (NoSuchAlgorithmException | NoSuchPaddingException |
InvalidKeyException |
                 BadPaddingException | IllegalBlockSizeException e) {
            throw new RuntimeException(e);
        }
    }
}
```

Листинг 6. Класс активности

```
public class MyLoader extends AsyncTaskLoader<String> {
    private String firstName;
    public static final String ARG_WORD = "word";
    public static final String ARG_CIPHER = "cipher";
    public static final String ARG_KEY = "Key";

    private Bundle args;


    public MyLoader(Context context, Bundle args) {
        super(context);
        if(args != null)
            this.firstName = args.getString(ARG_WORD);
            this.args = args;
    }
    @Override
    protected void onStartLoading() {
        super.onStartLoading();
        forceLoad();
    }
    @Override
    public String loadInBackground() {
        //SystemClock.sleep(5000);
        //return firstName;

        byte[] cryptText = args.getByteArray(ARG_WORD);
        byte[] key = args.getByteArray(ARG_KEY);
        SecretKey OriginKey = new SecretKeySpec(key, 0, key.length, "AES");
        return decryptMsg(cryptText, OriginKey);
    }

    public static String decryptMsg(byte[] cipherText, SecretKey secretKey) {
        try{
            Cipher cipher = Cipher.getInstance("AES");
            cipher.init(Cipher.DECRYPT_MODE, secretKey);
            return new String(cipher.doFinal(cipherText));

        }catch (NoSuchAlgorithmException | NoSuchPaddingException |
IllegalBlockSizeException |
                BadPaddingException | InvalidKeyException e){
            throw new RuntimeException(e);
        }
    }
}
```

Листинг 7. Класс «MyLoader»

В модуле «ServiceApp» была реализована функция воспроизведения музыки через фоновую службу. При нажатии на кнопку появляется сообщение и начинает играть мызка, которая продолжает играть после сворачивания приложения. При нажатии на другую кнопку служба останавливается (см. рис. 9 и листинг 8).

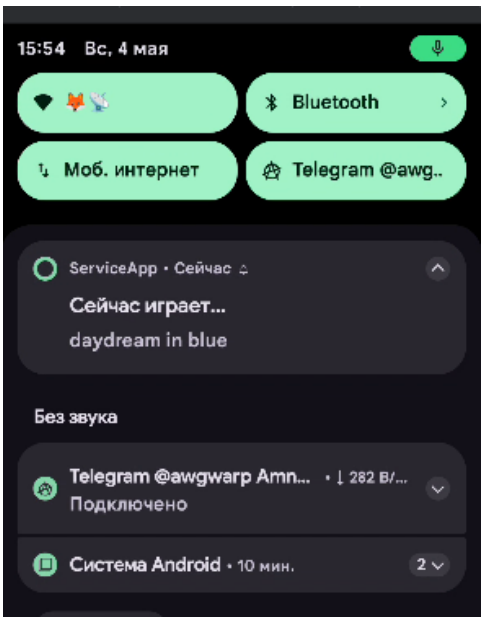

Рисунок 9. Сообщение о проигрывании музыки

```
public class PlayerService extends Service {
    private MediaPlayer mediaPlayer;
    public static final String CHANNEL_ID = "ForegroundServiceChannel";

    @Override
    public IBinder onBind(Intent intent) {
        // TODO: Return the communication channel to the service.
        throw new UnsupportedOperationException("Not yet implemented");
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        mediaPlayer.start();
        mediaPlayer.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
            @Override
            public void onCompletion(MediaPlayer mp) {
                stopForeground(true);
            }
        });
        return super.onStartCommand(intent, flags, startId);
    }



    @SuppressLint("ForegroundServiceType")
    @Override
    public void onCreate() {
        super.onCreate();
        NotificationCompat.Builder builder = new
NotificationCompat.Builder(this, CHANNEL_ID)
                .setContentText("Playing daydream in blue")
                .setSmallIcon(R.mipmap.ic_launcher)
                .setPriority(NotificationCompat.PRIORITY_HIGH)
                .setStyle(new
NotificationCompat.BigTextStyle().bigText("daydream in blue"))
                .setContentTitle("Сейчас играет...");
        int importance = NotificationManager.IMPORTANCE_DEFAULT;
        NotificationChannel channel = null;
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            channel = new NotificationChannel(CHANNEL_ID, "Aleev AV
```

```
Notification", importance);
        }
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            channel.setDescription("MIREA Channel");
        }
        NotificationManagerCompat notificationManager =
NotificationManagerCompat.from(this);
        assert channel != null;
        notificationManager.createNotificationChannel(channel);
        startForeground(1, builder.build());

        mediaPlayer = MediaPlayer.create(this, R.raw.daydreaminblue);
        mediaPlayer.setLooping(false);



    }
    @Override
    public void onDestroy() {
        stopForeground(true);
        mediaPlayer.stop();

    }
}
```

Листинг 8. Класс «PlayerService»

В следующем модуле «WorkManager» был добавлен критерий запуска «worker». Если устройство не подключено к wifi и к зарядке, но фоновый процесс не будет запущен (см. рис. 10 и листинг 9).

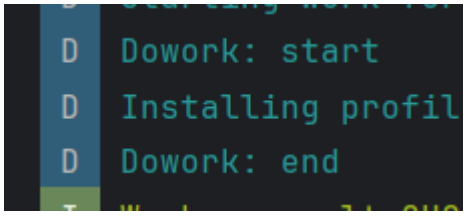

Рисунок 10. Пример выполнения процесса

```
public class MainActivity extends AppCompatActivity {

    private ActivityMainBinding binding;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);

        binding = ActivityMainBinding.inflate(getLayoutInflater());
        View view = binding.getRoot();
        setContentView(view);


        Constraints constraints = new Constraints.Builder()
                .setRequiredNetworkType(NetworkType.UNMETERED)
                        .setRequiresCharging(true)
                                .build();

        WorkRequest uploadWorkRequestt = new
OneTimeWorkRequest.Builder(UploadWorker.class)
                .setConstraints(constraints)
```

```
                        .build();

        WorkManager
                .getInstance(this)
                .enqueue(uploadWorkRequestt);

        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main),
(v, insets) -> {
            Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
            return insets;
        });
    }
}
```

Листинг 9. Выставление ограничений запуска процесса

В проект «MireaProject» был добавлен ещё один фрагмент, который позволяет включать музыку через сервис, как это было реализовано в модуле «ServiceApp» (см. рис. 11 и листинг 10).

Рисунок 11. Музыкальный плеер

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    // Inflate the layout for this fragment
```

```
    if (ContextCompat.checkSelfPermission(getActivity(), POST_NOTIFICATIONS)
== PackageManager.PERMISSION_GRANTED) {
        Log.d(MainActivity.class.getSimpleName().toString(), "Разрешения
получены");
    } else {
        Log.d(MainActivity.class.getSimpleName().toString(), "Нет
разрешений!");
        ActivityCompat.requestPermissions(getActivity(), new
String[]{POST_NOTIFICATIONS, FOREGROUND_SERVICE}, PermissionCode);


    }

    binding = FragmentPlayerBinding.inflate(getLayoutInflater());
    View view = binding.getRoot();

    binding.IButtonPlay.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Log.d(MainActivity.class.getSimpleName().toString(), "Кнопка
нажата");
            Intent service = new Intent(getActivity(), PlayerService.class);
            ContextCompat.startForegroundService(getActivity(), service);
        }
    });

    binding.IButtonStop.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            getActivity().stopService(new Intent(getActivity(),
PlayerService.class));
        }
    });
    return view;

}
```

Листинг 10. Код фрагмента