Разработка мобильных приложений

Практическая работа №6

Выполнил: Алеев А.В. БСБО-07-22

В ходе данной работы был создан проект «ru.mirea.AleevAV.Lesson6», в котором были созданы следующие модули «app», «employeedb», «internalfilestorage», «notebook» и «securesharedpreferences» (см. рис .1).

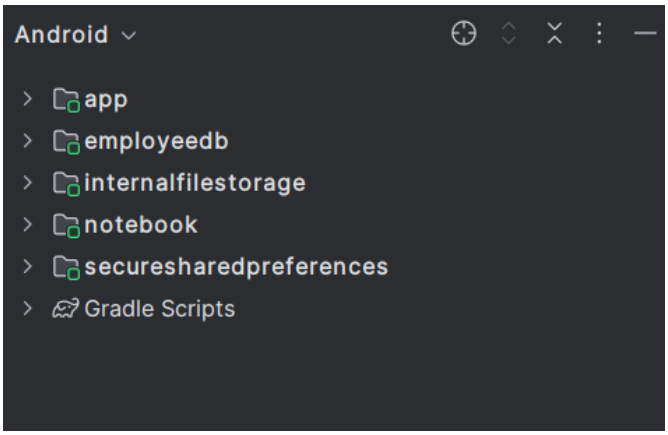

Рисунок 1. Модули проекта

В первом модуле «app» было создано 3 поля для ввода. При нажатии на кнопку «Отправить» данные сохраняются на устройстве (см. рис. 2,3 и листинг 1).

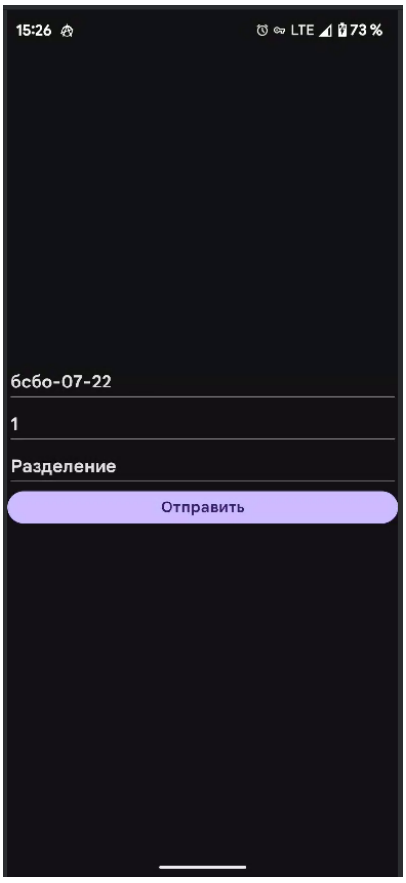

Рисунок 2. Окно приложения

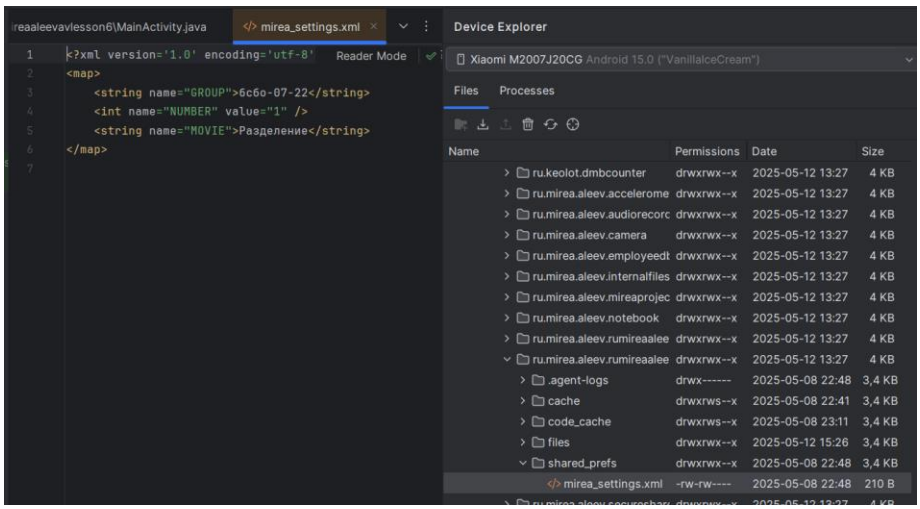

Рисунок 3. Сохранённые данные

```
buttonSave.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v){
        String group = editTextGroup.getText().toString();
        int number = Integer.parseInt(editTextList.getText().toString());
        String movie = editMovie.getText().toString();

        SharedPreferences.Editor editor = sharedPreferences.edit();
        editor.putString("GROUP", group);
        editor.putInt("NUMBER", number);
        editor.putString("MOVIE", movie);
        editor.apply();

        Toast.makeText(MainActivity.this, "Данные сохранены!",
Toast.LENGTH_SHORT).show();
    }
```

Листинг 1. Метод для сохранения

Далее был создан модуль «securesharedpreferences», в котором был создан экран отображения имени и фото писателя. При нажатии на кнопку «Сохранить данные» данные сохраняются на устройстве через secureSharedPreferences (см. рис. 4, 5 и листинг 2).

Рисунок 4. Экран модуля

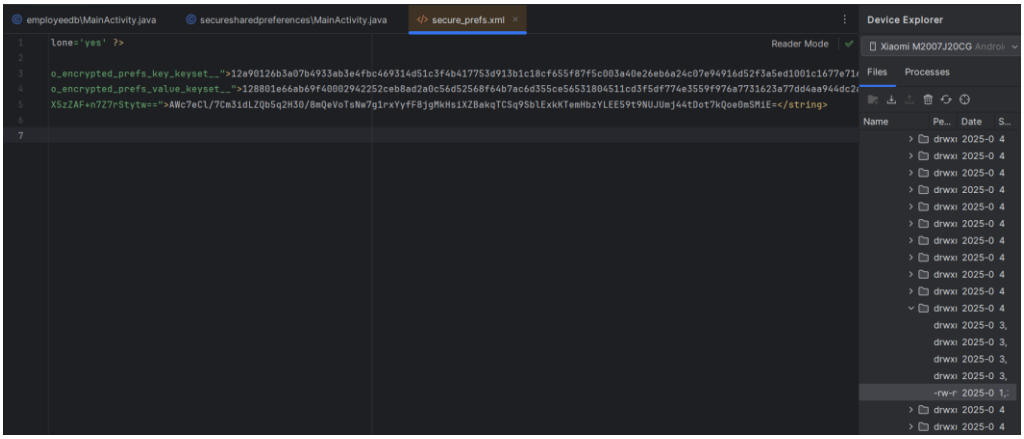

Рисунок 5. Пример данных

```
public class MainActivity extends AppCompatActivity {
    private ActivityMainBinding binding;
    private TextView textView;
    private SharedPreferences secureSharedPreferences;


    @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());

        textView = binding.Maintext;
        Button buttonSave = binding.button;

        try {
            String masterKey =
MasterKeys.getOrCreate(MasterKeys.AES256_GCM_SPEC);
            secureSharedPreferences = EncryptedSharedPreferences.create(
                    "secure_prefs",
                    masterKey,
                    this,

EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256_SIV,

EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256_GCM
            );
        } catch (GeneralSecurityException | IOException e) {
            throw new RuntimeException(e);
        }

        String WritterName = secureSharedPreferences.getString("WRITE_NAME",
"Писатель: Филип К. Дик");
        textView.setText(WritterName);

        buttonSave.setOnClickListener(v ->{
            secureSharedPreferences.edit().putString("WRITE_NAME",
```

```
textView.getText().toString()).apply();
            Toast.makeText(this, "Данные сохранены!",
Toast.LENGTH_SHORT).show();
        });

    }
}
```

Листинг 2. Код модуля

Далее был создан модуль «InternalFileStorage», в котором был создан экран для сохранения данных в «txt» файле (см. рис. 6 и листинг 3)

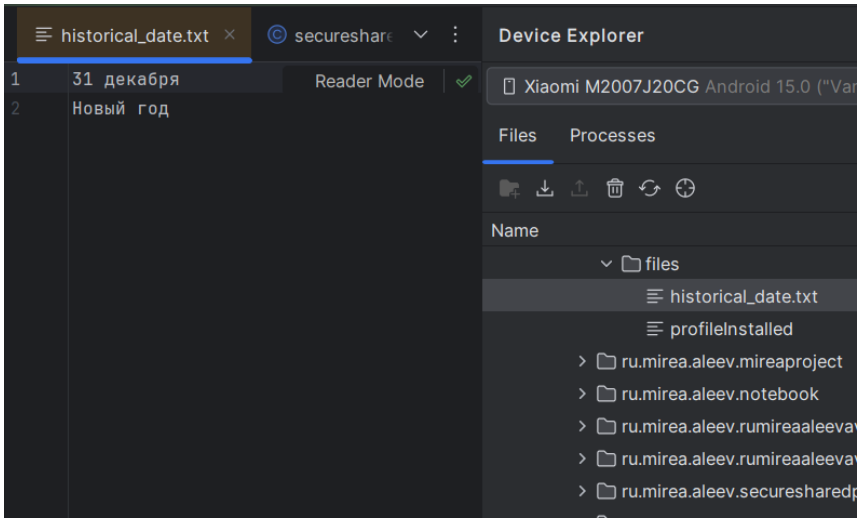

Рисунок 6. Пример сохранённых данных

```
public class MainActivity extends AppCompatActivity {

    private EditText editTextDate, editTextDescription;
    private static final String FILENAME = "historical_date.txt";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        editTextDate = findViewById(R.id.editTextDate);
        editTextDescription = findViewById(R.id.editTextDesc);
        Button buttonSave = findViewById(R.id.button);

        buttonSave.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String date = editTextDate.getText().toString();
                String description =
editTextDescription.getText().toString();
                String data = date + "\n" + description;

                // Запись в файл
                try (FileOutputStream fos = openFileOutput(FILENAME,
MODE_PRIVATE)) {
                    fos.write(data.getBytes());
                    Toast.makeText(MainActivity.this, "Файл сохранён!",
Toast.LENGTH_SHORT).show();
                } catch (IOException e) {
                    Toast.makeText(MainActivity.this, "Ошибка записи!",
Toast.LENGTH_SHORT).show();
```

```
                    e.printStackTrace();
                }
            }
        });
    }
}
```

Листинг 3. Код для сохранения в «txt» файле

Далее был создан модуль «Notebook», в котором было реализовано сохранение данных в файле и загрузка данных (см. рис. 7 и листинг 4).

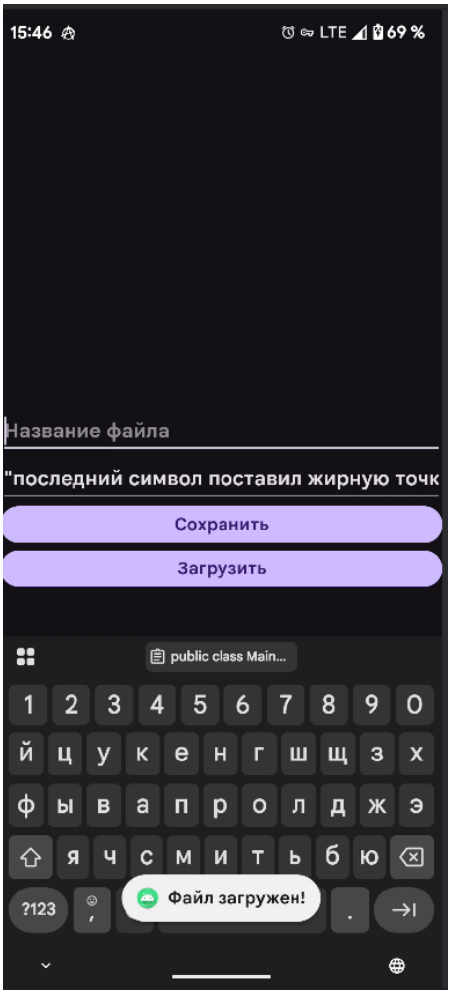

Рисунок 7 Пример загрузки файла

```
public class MainActivity extends AppCompatActivity {
    private EditText editTextFileName, editTextQuote;
    private ActivityResultLauncher<Intent> saveFileLauncher;
    private ActivityResultLauncher<Intent> loadFileLauncher;
    private Uri currentFileUri;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        editTextFileName = findViewById(R.id.editTextFileName);
        editTextQuote = findViewById(R.id.editTextQuote);
        Button buttonSave = findViewById(R.id.buttonSave);
        Button buttonLoad = findViewById(R.id.buttonLoad);
```

```
        // Инициализация лаунчеров для выбора файлов
        saveFileLauncher = registerForActivityResult(
                new ActivityResultContracts.StartActivityForResult(),
                result -> {
                    if (result.getResultCode() == Activity.RESULT_OK &&
result.getData() != null) {
                        Uri uri = result.getData().getData();
                        if (uri != null) {
                            writeFileToUri(uri);
                        }
                    }
                }
        );

        loadFileLauncher = registerForActivityResult(
                new ActivityResultContracts.StartActivityForResult(),
                result -> {
                    if (result.getResultCode() == Activity.RESULT_OK &&
result.getData() != null) {
                        Uri uri = result.getData().getData();
                        if (uri != null) {
                            readFileFromUri(uri);
                        }
                    }
                }
        );

        // Сохранение файла через системный диалог
        buttonSave.setOnClickListener(v -> {
            String fileName = editTextFileName.getText().toString().trim();
            if (fileName.isEmpty()) {
                Toast.makeText(this, "Введите название файла!",
Toast.LENGTH_SHORT).show();
                return;
            }

            Intent intent = new Intent(Intent.ACTION_CREATE_DOCUMENT);
            intent.addCategory(Intent.CATEGORY_OPENABLE);
            intent.setType("text/plain");
            intent.putExtra(Intent.EXTRA_TITLE, fileName + ".txt");
            saveFileLauncher.launch(intent);
        });

        // Загрузка файла через системный диалог
        buttonLoad.setOnClickListener(v -> {
            Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);
            intent.addCategory(Intent.CATEGORY_OPENABLE);
            intent.setType("text/plain");
            loadFileLauncher.launch(intent);
        });
    }

    private void writeFileToUri(Uri uri) {
        try {
            String quote = editTextQuote.getText().toString();
            OutputStream outputStream =
getContentResolver().openOutputStream(uri);
            outputStream.write(quote.getBytes());
            outputStream.close();
            Toast.makeText(this, "Файл сохранён!",
Toast.LENGTH_SHORT).show();
        } catch (Exception e) {
            Toast.makeText(this, "Ошибка записи!",
Toast.LENGTH_SHORT).show();
```

```
        }
    }

    private void readFileFromUri(Uri uri) {
        try {
            InputStream inputStream =
getContentResolver().openInputStream(uri);
            BufferedReader reader = new BufferedReader(new
InputStreamReader(inputStream));
            StringBuilder stringBuilder = new StringBuilder();
            String line;
            while ((line = reader.readLine()) != null) {
                stringBuilder.append(line).append("\n");
            }
            editTextQuote.setText(stringBuilder.toString().trim());
            Toast.makeText(this, "Файл загружен!",
Toast.LENGTH_SHORT).show();
        } catch (Exception e) {
            Toast.makeText(this, "Ошибка чтения!",
Toast.LENGTH_SHORT).show();
        }
    }
}
```

Листинг 4. Класс для сохранения и загрузки файлов

Затем был создан модуль «EmployeeDB», в котором была реализовано база данных для хранения информации о вымышленных супер-героях (см. листинг 5,6,7 и 8).

```
public class MainActivity extends AppCompatActivity {
    private AppDatabase db;
    private SuperHeroDao superHeroDao;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Инициализация базы данных
        db = App.getInstance().getDatabase();
        superHeroDao = db.superHeroDao();

        // Добавление супер-героя
        SuperHero hero = new SuperHero("Человек-паук", "Паучьи способности",
10);
        superHeroDao.insert(hero);

        // Получение всех героев
        List<SuperHero> heroes = superHeroDao.getAll();
        Toast.makeText(this, "Сохранено героев: " + heroes.size(),
Toast.LENGTH_SHORT).show();
    }
}
```

Листинг 5. Класс основного экрана

```
@Entity(tableName = "superheroes")
public class SuperHero {
    @PrimaryKey(autoGenerate = true)
    public int id;
    public String name;
    public String power;
```

```
    public int level;

    public SuperHero(String name, String power, int level) {
        this.name = name;
        this.power = power;
        this.level = level;
    }
}
```

Листинг 6. Класс «SuperHero»

```
public abstract class AppDatabase extends RoomDatabase {
    public abstract SuperHeroDao superHeroDao();

    private static volatile AppDatabase INSTANCE;

    public static AppDatabase getInstance(Context context) {
        if (INSTANCE == null) {
            synchronized (AppDatabase.class) {
                if (INSTANCE == null) {
                    INSTANCE = Room.databaseBuilder(
                            context.getApplicationContext(),
                            AppDatabase.class,
                            "superhero_db"
                    ).allowMainThreadQueries().build();
                }
            }
        }
        return INSTANCE;
    }
}
```

Листинг 7. Класс «AppDatabase»

```
public class App extends Application {
    private static App instance;
    private AppDatabase database;

    @Override
    public void onCreate() {
        super.onCreate();
        instance = this;
        database = AppDatabase.getInstance(this);
    }

    public static App getInstance() {
        return instance;
    }

    public AppDatabase getDatabase() {
        return database;
    }
}
```

Листинг 8. Класс «App»

Далее в проекте «MireaProject» было добавлено 2 фрагмента.

В первом фрагменте «Профиль» было реализовано сохранение данных (см. рис. 8)

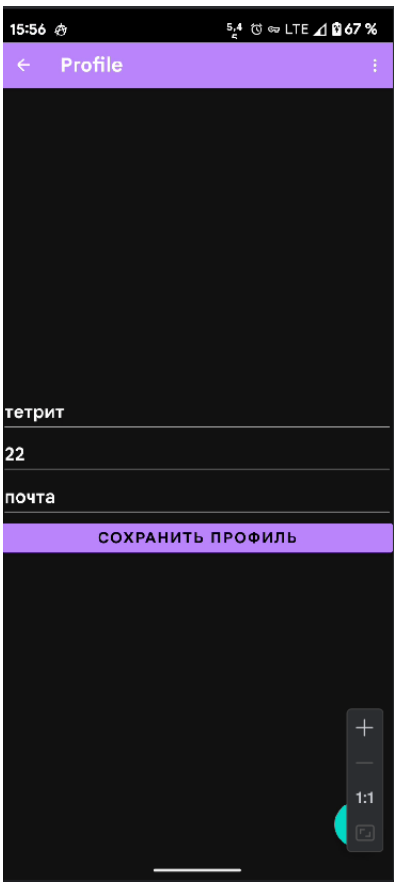

Рисунок 8. Пример «профиля»

Во втором фрагменте была реализован функционал, связанный с обработкой файлов. При нажатии на кнопку «+» и выборе «шифровать файл» шифруется время и сохраняется на устройстве (см рис. 9).

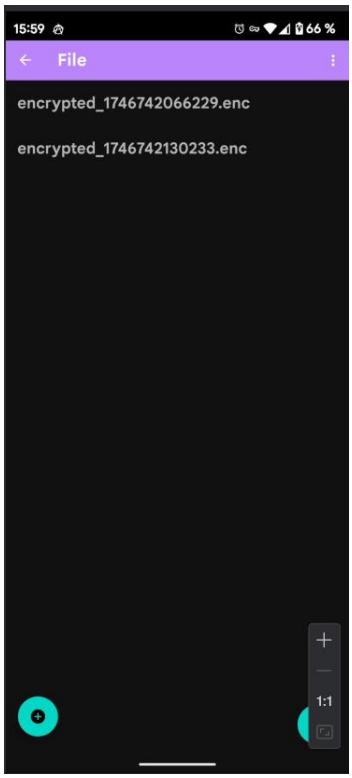

Рисунок 9. Фрагмент для шифрования времени