Лабораторная работа № 8

«Работа с фрагментами в Android-приложении»

Продолжительность выполнения лабораторной работы: 2 ак. часа.

Целью лабораторной работы является знакомство с принципами создания и использования фрагментов в Android-приложениях.

Практическая часть

Создайте новый проект **FragmentsApp** с пустой активити.

Для работы с фрагментами потребуется библиотека **AndroidX Fragment library**. Что подключить данную библиотеку необходимо добавить репозиторий Google Maven в файл settings.gradle вашего проекта, а затем добавить определенные зависимости в файл build.gradle.

Gradle — это система автоматической сборки, построенная на принципах Apache Ant (утилита для автоматизации процесса сборки программного продукта) и Apache Maven (фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM, являющемся подмножеством XML), которая используется в Android Studio.

Файл settings.gradle может состоять из одной строчки: include ':app'

Это означает, что используется один проект для работы.

Также репозиторий Google Maven может быть добавлен в файл settings.gradle автоматически. Необходимо проверить, чтобы данный файл содержал код, представленный в листинге 1.

Листинг 1. Фрагмент settings.gradle

```
dependencyResolutionManagement {
repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
    repositories {
        google()
        mavenCentral()
    }
}
```

В файл build.gradle вашего приложения (Module: FragmentsApp.app) в секцию dependencies добавляем зависимость "androidx.fragment:fragment: 1.4.1" (листинг 2).

Листинг 2. Фрагмент build.gradle (Module: FragmentsApp.app)

```
dependencies {
    implementation "androidx.fragment:fragment:1.4.1"
...
```

Затем следует нажать на ссылку «Sync Now», которая появится в предупреждении (рисунок 1).

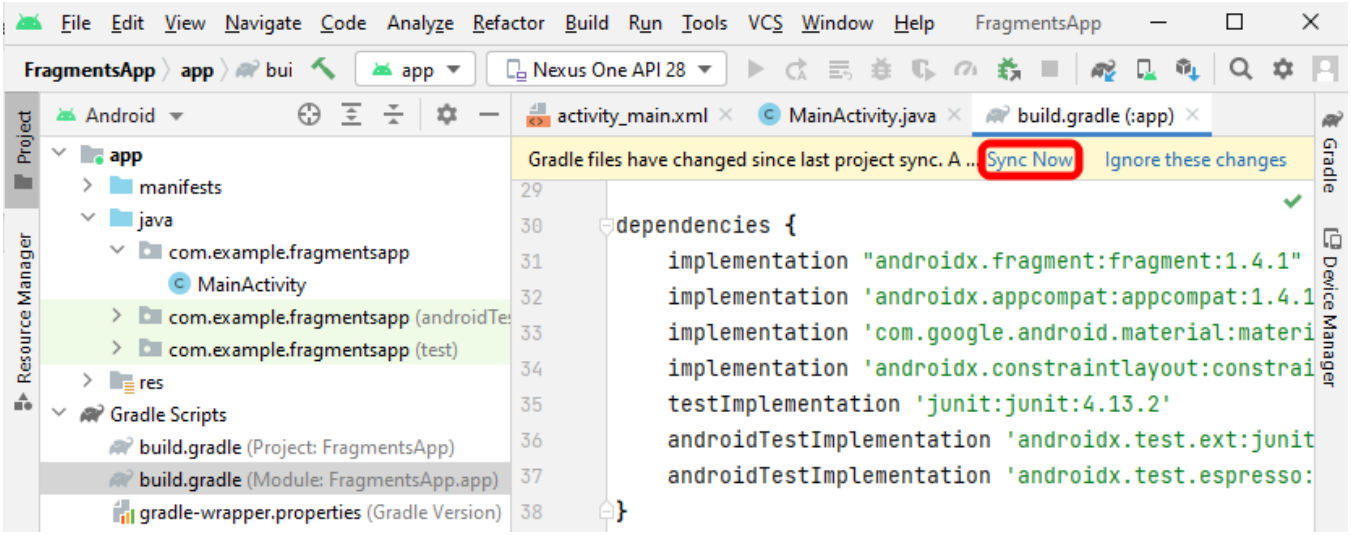

Рисунок 1. Синхронизация файлов Gradle.

Разметка интерфейса фрагмента может быть описана в xml-файле, так же как для активити. Создадим файл fragment_detail.xml (листинг 3) в папке res/layout с разметкой для фрагмента DetailFragment, в котором в разрабатываемом приложении будет отображаться описание определённого объекта при выборе из списка в фрагменте ListFragment (в примере будет использоваться список из видов цветов).

Листинг 3. Файл fragment_detail.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
    <ImageView
        android:id="@+id/photo"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:padding="5dp"/>
    <ScrollView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
    <TextView
        android:id="@+id/detailsText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Выберите цветок"
```

```
        android:textSize="12sp"
        android:padding="5dp"/>
    </ScrollView>
</LinearLayout>
```

Также необходимо создать файл fragment_list.xml (листинг 4) в папке res/layout с разметкой для фрагмента ListFragment, в котором, будет содержаться ListView для вывода списка объектов.

Листинг 4. Файл fragment_list.xml

```
<ListView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="10dp"
    android:id="@+id/list"/>
```

Таким образом содержимое фрагментов будет очень простым: один фрагмент будет содержать список объектов (например, цветов), а второй — текстовой поле с описанием объекта и его изображение. При выборе элемента в списке в фрагменте ListFragment описание выбранного объекта и фотография отобразится во фрагменте DetailFragment.

Также в файле компоновки activity_main.xml необходимо создать два элемента FragmentContainerView для того, чтобы выводить внутри этих элементов фрагменты (листинг 5).

Листинг 5. Файл activity_main.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <androidx.fragment.app.FragmentContainerView
        android:id="@+id/listFragment"
        android:layout_width="match_parent"
        android:layout_height="300dp"
        android:name="com.example.fragmentsapp.ListFragment" />
    <androidx.fragment.app.FragmentContainerView
        android:id="@+id/detailFragment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:name="com.example.fragmentsapp.DetailFragment"/>
</LinearLayout>
```

Теперь добавим в проект в одну папку с MainActivity собственно классы фрагментов. Добавим новый класс ListFragment (листинг 6).

Фрагменты не могут напрямую взаимодействовать между собой. Для этого надо обращаться к контексту, в качестве которого выступает класс Activity. Для обращения к активити, как правило, создается вложенный интерфейс. В данном случае он называется OnFragmentSendDataListener с одним методом onSendData().

Но чтобы взаимодействовать с другим фрагментом через активити, нам надо прикрепить текущий фрагмент к активити. Для этого в классе фрагмента определен метод onAttach(Context context). В нем происходит установка объекта OnFragmentSendDataListener:

fragmentSendDataListener = (OnFragmentSendDataListener) context;

При обработке нажатия на элемент в списке мы можем отправить Activity данные о выбранном объекте:

String selectedItem = (String)parent.getItemAtPosition(position);

fragmentSendDataListener.onSendData(selectedItem);

Таким образом, при выборе объекта в списке MainActivity получит выбранный объект.

Листинг 6. Класс ListFragment

```
package com.example.fragmentsapp;
import android.content.Context;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import androidx.fragment.app.Fragment;

public class ListFragment extends Fragment {
    interface OnFragmentSendDataListener {
        void onSendData(String data);
    }
    private OnFragmentSendDataListener fragmentSendDataListener;
    String[] flowers = { "Ветреница дубравная (Anemone nemorosa)",
            "Горянка (Epimedium)",
            "Зубянка железистая (Cardamine glanduligera)",
            "Морозник (Helleborus)",
            "Печёночница (Hepatica)",
            "Сердечник луговой (Cardamine pratensis)",
            "Триллиум (Trillium)"};
    @Override
```

```
    public void onAttach(Context context) {
        super.onAttach(context);
        try {
            fragmentSendDataListener = (OnFragmentSendDataListener)
context;
        } catch (ClassCastException e) {
            throw new ClassCastException(context.toString()
                    + " должен реализовывать интерфейс
OnFragmentInteractionListener");
        }
    }
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_list, container,
false);
        // получаем элемент ListView
        ListView flowersList = view.findViewById(R.id.list);
        // создаем адаптер
        ArrayAdapter<String> adapter = new ArrayAdapter(getContext(),
android.R.layout.simple_list_item_1, flowers);
        // устанавливаем для списка адаптер
        flowersList.setAdapter(adapter);
        // добавляем для списка слушатель
        flowersList.setOnItemClickListener(new
AdapterView.OnItemClickListener(){

            @Override
            public void onItemClick(AdapterView<?> parent, View v, int
position, long id)
            {
                // получаем выбранный элемент
                String selectedItem =
(String)parent.getItemAtPosition(position);
                // Посылаем данные Activity
                fragmentSendDataListener.onSendData(selectedItem);
            }
        });
        return view;
    }
}
```

Теперь определим класс для второго фрагмента DetailFragment (листинг 7).

Задача этого фрагмента – вывод некоторой информации. Так как он не должен передавать никакую информацию другому фрагменту, здесь мы можем ограничиться

только переопределением метода onCreateView(), который в качестве визуального интерфейса устанавливает разметку из файла fragment_detail.xml

Но чтобы имитировать взаимодействие между двумя фрагментами, здесь также определен метод setSelectedItem(), который обновляет текст в текстовом поле и изображение.

Листинг 7. Класс DetailFragment

```
...
public class DetailFragment extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container, Bundle savedInstanceState) {
    return inflater.inflate(R.layout.fragment_detail, container, false);
    }

    public void setSelectedItem(String selectedItem) {
        TextView desc = getView().findViewById(R.id.detailsText);
        ImageView photo = getView().findViewById(R.id.photo);
        switch (selectedItem) {
            case "Ветреница дубравная (Anemone nemorosa)":
                desc.setText(R.string.Anemone_desc);
                photo.setImageResource(R.drawable.anemone);
                break;
            case "Горянка (Epimedium)":
                desc.setText(R.string.Epimedium_desc);
                photo.setImageResource(R.drawable.epimedium);
                break;
            ...
        }
    }
}
```

С помощью двух элементов компоновки FragmentContainerView в MainActivity добавляются два выше определенных фрагмента.

Листинг 8. Класс MainActivity

```
...
public class MainActivity extends AppCompatActivity
        implements ListFragment.OnFragmentSendDataListener {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    @Override
    public void onSendData(String selectedItem) {
```

```
        DetailFragment fragment = (DetailFragment)
getSupportFragmentManager()
                .findFragmentById(R.id.detailFragment);
        if (fragment != null)
            fragment.setSelectedItem(selectedItem);
    }
}
```

Для взаимодействия фрагмента ListFragment с другим фрагментом через MainActivity надо, чтобы эта activity реализовывала интерфейс OnFragmentSendDataListener. Для этого реализуем метод onSendData, который получает фрагмент DetailFragment и вызывает у него метод setSelectedItem.

В итоге получится, что при выборе в списке во фрагменте ListFragment будет срабатывать слушатель списка и, в частности, его метод onItemClick, который вызовет метод fragmentSendDataListener.onSendData(selectedItem). fragmentSendDataListener устанавливается как MainActivity, поэтому при этом будет вызван метод setSelectedItem у фрагмента DetailFragment. Таким образом, произойдет взаимодействие между двумя фрагментами.

Для более удобного расположения при альбомной ориентации, как правило, решение довольно простое – два фрагмента располагаются горизонтально в ряд.

Для этого необходимо создать папку res/layout-land и разместить в ней файл activity_main.xml с компоновкой для альбомной ориентации.

Задание

Разработайте приложение, аналогичное описанному в практической части, т.е. приложение FragmentsApp с двумя фрагментами: один фрагмент будет содержать список неких объектов, а второй — текстовой поле с описанием объекта и его изображение. При выборе элемента из списка в фрагменте ListFragment описание выбранного объекта и фотография отображается во фрагменте DetailFragment. Разработайте различные компоновки для альбомной и портретной ориентаций экрана.

Контрольные вопросы:

1. Что такое фрагменты?
2. Когда вызывается метод onCreateView?
3. Для чего используется FragmentManager?
4. Какой метод содержится в интерфейсе OnFragmentSendDataListener?
5. Для чего используется метод findFragmentById?