МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИИТ)

Кафедра математического обеспечения и стандартизации ИТ

ОТЧЁТ ПО ПРАКТИЧЕСКОМУ ЗАНЯТИЮ №5

по дисциплине

«Разработка мобильных приложений»

Отчет представлен к

рассмотрению:

Студенты группы ИНБО-04-20 «28» февраля 2022 г. _____ Ло В.Х.

(подпись)

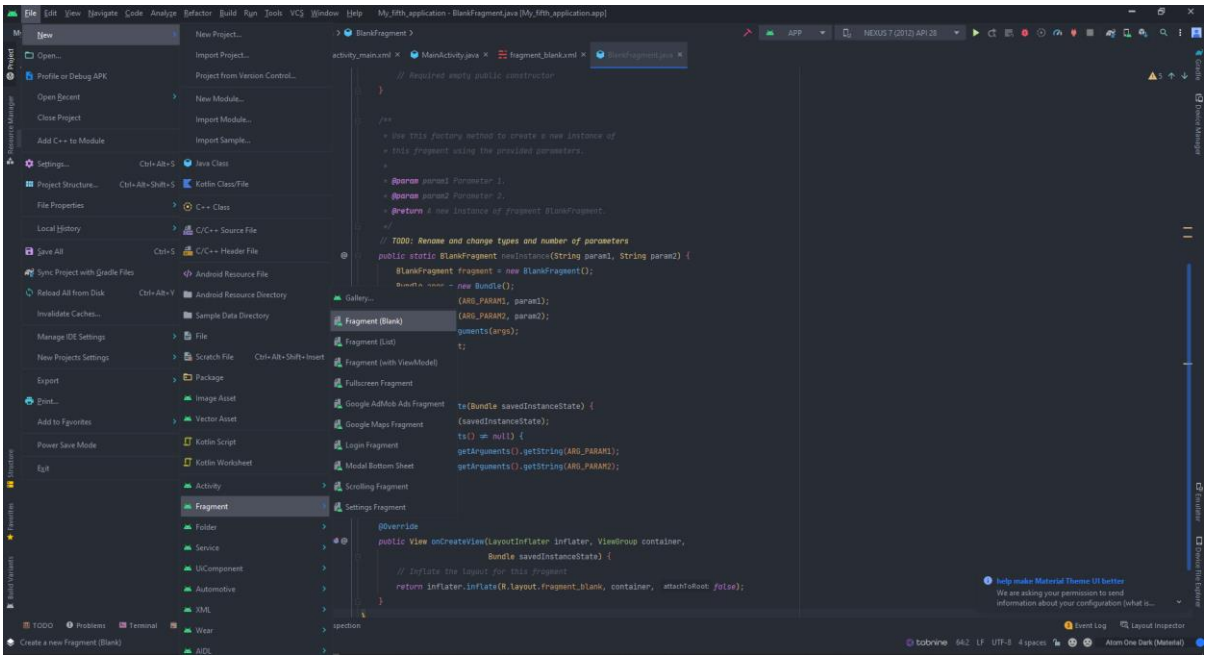
Преподаватель « » 2022 г. _____ Фандеев И.И.

(подпись)

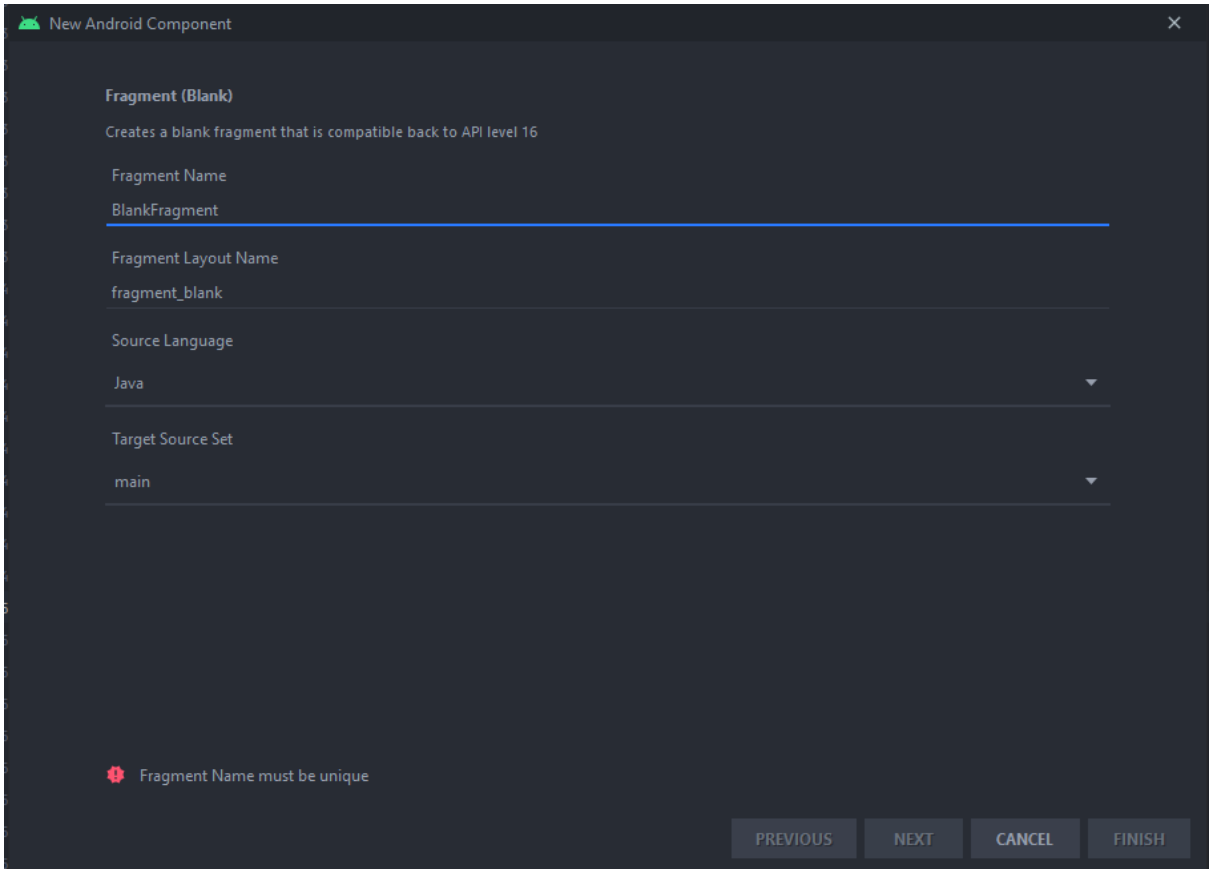
Москва, 2022г.

1. Реализовать создание класса фрагмента

Чтобы создать fragment, мы выбираем папку с именем java, затем щелкаем правой кнопкой мыши, выбираем «New», затем выбираем «Fragment», затем выбираем «Fragment (blank)».



Затем мы назовем Fragment, а затем нажмите «Finish», чтобы закончить.



Вот результат

```java
package com.example.my_fifth_application;

import android.os.Bundle;

import androidx.fragment.app.Fragment;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

/**
 * A simple {@link Fragment} subclass.
 * Use the {@link BlankFragment#newInstance} factory method to
 * create an instance of this fragment.
 */
public class BlankFragment extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_blank, container, attachToRoot: false);
    }
```

2. Реализовать добавление фрагмента в XML разметку явлений

После создания фрагмента, как указано выше, мы получаем файл fargment_first.XML следующим образом. Приступаем к добавлению фрагмента в XML-разметку событий

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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=".FirstFragment">

    <fragment
        android:name="com.example.my_fifth_application.FirstFragment"
        android:id="@+id/first_fragment"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="1"/>
    <fragment
        android:name="com.example.my_fifth_application.SecondFragment"
        android:id="@+id/second_fragment"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="2"/>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="Hello blank fragment" />

</LinearLayout>
```

Затем мы отредактируем файл MainActivity.java следующим образом.

```java
package com.example.my_fifth_application;

import ...

public class MainActivity extends FragmentActivity {

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

3. Реализовать создание гибкого интерфейса пользователя

Гибкие пользовательские интерфейсы являются ключевой частью правильной разработки приложений для Android. Нам нужно убедиться, что наши приложения работают на самых разных устройствах, версиях, ориентациях и локалях. К счастью, в Android есть несколько мощных конструкций, которые делают отзывчивые, гибкие пользовательские интерфейсы довольно простыми. Обычно это используется для создания надежных приложений, которые работают в различных условиях.

4. Реализовать добавление фрагмента в явление во время выполнения

Чтобы добавить фрагмент к действию во время выполнения, мы должны изменить файл fragment_first.XML следующим образом.

```xml
<?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:id="@+id/fragment_container"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

```

Мы изменим файл MainActivity.java следующим образом.

```java
package com.example.my_fifth_application;

import ...

public class MainActivity extends FragmentActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.fragment_first);
        if (findViewById(R.id.fragment_container) != null) {
            if (savedInstanceState != null) {
                return;
            }
            FirstFragment fFagment = new FirstFragment();
            fFagment.setArguments(getIntent().getExtras());
            getSupportFragmentManager().beginTransaction().add(R.id.fragment_container, fFagment).commit();
        }
    }
```

5. Реализовать замену одного фрагмента другим

Для замены одного фрагмента другим изменим в файле MainActivity.java следующим образом

```java
FirstFragment newFrag = new FirstFragment();
Bundle args = new Bundle();
args.putInt(FirstFragment.ARG_POSITION, position);
newFrag.setArguments(args);
FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();
transaction.replace(R.id.fragment_container,newFrag);
transaction.addToBackStack(null);
transaction.commit();
```

6. Реализовать определение интерфейса взаимодействия фрагмента с явлением

Для определения интерфейса того, как фрагмент взаимодействует с явлением, мы изменим файл FirstFagment.java следующим образом.

```java
public class FirstFragment extends Fragment {
    OnFirstSelectedListener mCallback;

    public interface OnFirstSelectedListener {
        public void onSecondSelected(int position);
    }


    public void onAttach(Activity activity) {
        super.onAttach(activity);
        try {
            mCallback = (OnFirstSelectedListener) activity;
        } catch (ClassCastException e) {
            throw new ClassCastException(activity.toString() +"must implement OnFirstSelectedListener");
        }
    }
    public void onListItemClick(ListView l, View v, int position,long id){
        mCallback.onSecondSelected(position);
    }
```

7. Реализовать интерфейс взаимодействия

Чтобы реализовать интерфейс взаимодействия, мы изменим файл MainActivity.java следующим образом.

```java
public class MainActivity extends Activity implements FirstFragment.OnFirstSelectedListener {


    public void onActicleSelected(int position){

    }
```

8. Реализовать доставку сообщения фрагменту

Чтобы реализовать доставку сообщений во фрагмент, мы изменим файл MainActivity.java следующим образом.

```java
public static class MainActivity extends Activity implements FirstFragment.OnFirstSelectedListener {




    public void onSecondSelected(int position){
        SecondFragment sFrag = (SecondFragment).getSupportFragmentManager().findFragmentById(R.id.second_fragment);
        if (sFrag != null){
            sFrag.updateSecondView(position);
        }else{
            SecondFragment newFrag = new SecondFragment();
            Bundle args = new Bundle();
            args.putInt(SecondFragment.ARG_POSITION, position);
            newFrag.setArguments(args);
            FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();
            transaction.replace(R.id.fragment_container,newFrag);
            transaction.addToBackStack(null);
            transaction.commit();
        }
    }
```