|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

**Отчет по практической работе №8**

по дисциплине «Разработка мобильных приложений»

|  |  |
| --- | --- |
| **Выполнил:**  студент группыИКБО-28-22 | Некрасов Г.А. |
| **Проверил:**  к.э.н доцент | Петросян Л.Э. |

Москва 2024 г.

**СОДЕРЖАНИЕ**

[Задание 3](#_Toc167626991)

[Выполнение работы 3](#_Toc167626992)

[Тестирование работы 8](#_Toc167626993)

# Задание

* Реализовать запуск последовательно 3 задач.
* Реализовать запуск 2 задач параллельно.
* Разработать приложение, которое при нажатии кнопки будет загружать изображение по Rest API: [https://random.dog/woof.json](https://random.dog/woof.json%20) и выводить его на экран.

# Выполнение работы

Листинг 1. Файл activity\_main.xml

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?> <RelativeLayout 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=".MainActivity">   <Button  android:id="@+id/buttonSequential"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Запуск последовательно 3 задач"  android:layout\_centerHorizontal="true"  android:layout\_marginTop="50dp"/>   <Button  android:id="@+id/buttonParallel"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Запуск 2 задач параллельно"  android:layout\_below="@id/buttonSequential"  android:layout\_centerHorizontal="true"  android:layout\_marginTop="20dp"/>   <Button  android:id="@+id/buttonLoadImage"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Загрузить изображение"  android:layout\_below="@id/buttonParallel"  android:layout\_centerHorizontal="true"  android:layout\_marginTop="20dp"/>   <!-- Добавлен ImageView -->  <ImageView  android:id="@+id/imageView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@id/buttonLoadImage"  android:layout\_centerHorizontal="true"  android:layout\_marginTop="20dp"/>  </RelativeLayout> |

Листинг 2. Файл MainActivity.java

|  |
| --- |
| package com.example.nekrasovglebandreevich\_8pract;  import android.graphics.Bitmap; import android.graphics.BitmapFactory; import android.os.AsyncTask; import android.os.Bundle; import android.util.Log; import android.view.View; import android.widget.Button; import android.widget.ImageView; import android.widget.Toast; import androidx.appcompat.app.AppCompatActivity; import org.json.JSONException; import org.json.JSONObject; import java.io.BufferedReader; import java.io.IOException; import java.io.InputStream; import java.io.InputStreamReader; import java.net.HttpURLConnection; import java.net.URL;  public class MainActivity extends AppCompatActivity {   private ImageView imageView;  private static final String *TAG* = "MainActivity";  private boolean task3Executed = false; // Флаг для отслеживания выполнения Task3   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);   imageView = findViewById(R.id.*imageView*);   Button buttonSequential = findViewById(R.id.*buttonSequential*);  buttonSequential.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  executeSequentialTasks();  }  });   Button buttonParallel = findViewById(R.id.*buttonParallel*);  buttonParallel.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  executeParallelTasks();  }  });   Button buttonLoadImage = findViewById(R.id.*buttonLoadImage*);  buttonLoadImage.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  new LoadImageTask().execute();  }  });  }   private void executeSequentialTasks() {  // Запуск последовательно трех задач  new Task1().execute();  }   private void executeParallelTasks() {  // Запуск двух задач параллельно  new Task2().executeOnExecutor(AsyncTask.*THREAD\_POOL\_EXECUTOR*);  new Task3().executeOnExecutor(AsyncTask.*THREAD\_POOL\_EXECUTOR*);  }   private class Task1 extends AsyncTask<Void, Void, Void> {  @Override  protected Void doInBackground(Void... voids) {  // Задача 1  Log.*d*(*TAG*, "Task1: Execution");  // Можно добавить сюда код первой задачи   // Задержка для эмуляции работы  try {  Thread.*sleep*(1000);  } catch (InterruptedException e) {  e.printStackTrace();  }   return null;  }   @Override  protected void onPostExecute(Void aVoid) {  super.onPostExecute(aVoid);  Log.*d*(*TAG*, "Task1: Post-Execution");  // Выполнение следующей задачи  new Task2().execute();  }  }   private class Task2 extends AsyncTask<Void, Void, Void> {  @Override  protected Void doInBackground(Void... voids) {  // Задача 2  Log.*d*(*TAG*, "Task2: Execution");  // Можно добавить сюда код второй задачи   // Задержка для эмуляции работы  try {  Thread.*sleep*(1000);  } catch (InterruptedException e) {  e.printStackTrace();  }   return null;  }   @Override  protected void onPostExecute(Void aVoid) {  super.onPostExecute(aVoid);  Log.*d*(*TAG*, "Task2: Post-Execution");  // Выполнение следующей задачи, если Task3 еще не был выполнен  if (!task3Executed) {  task3Executed = true; // Устанавливаем флаг выполнения Task3  new Task3().execute();  }  }  }   private class Task3 extends AsyncTask<Void, Void, Void> {  @Override  protected Void doInBackground(Void... voids) {  // Задача 3  Log.*d*(*TAG*, "Task3: Execution");  // Можно добавить сюда код третьей задачи   // Задержка для эмуляции работы  try {  Thread.*sleep*(1000);  } catch (InterruptedException e) {  e.printStackTrace();  }   return null;  }   @Override  protected void onPostExecute(Void aVoid) {  super.onPostExecute(aVoid);  Log.*d*(*TAG*, "Task3: Post-Execution");  }  }   private class LoadImageTask extends AsyncTask<Void, Void, Bitmap> {  @Override  protected Bitmap doInBackground(Void... voids) {  try {  // Получение ссылки на изображение по Rest API  URL apiUrl = new URL("https://random.dog/woof.json");  HttpURLConnection connection = (HttpURLConnection) apiUrl.openConnection();  connection.connect();   InputStream inputStream = connection.getInputStream();  BufferedReader reader = new BufferedReader(new InputStreamReader(inputStream));  StringBuilder response = new StringBuilder();  String line;  while ((line = reader.readLine()) != null) {  response.append(line);  }  reader.close();   // Извлечение ссылки на изображение из JSON  JSONObject jsonObject = new JSONObject(response.toString());  String imageUrl = jsonObject.getString("url");   // Загрузка изображения по ссылке  URL url = new URL(imageUrl);  HttpURLConnection imageConnection = (HttpURLConnection) url.openConnection();  imageConnection.connect();  InputStream imageInputStream = imageConnection.getInputStream();  return BitmapFactory.*decodeStream*(imageInputStream);  } catch (IOException | JSONException e) {  e.printStackTrace();  return null;  }  }   @Override  protected void onPostExecute(Bitmap bitmap) {  if (bitmap != null) {  // Вывод изображения на экран  imageView.setImageBitmap(bitmap);  } else {  Toast.*makeText*(MainActivity.this, "Ошибка загрузки изображения", Toast.*LENGTH\_SHORT*).show();  }  Log.*d*(*TAG*, "LoadImageTask: Post-Execution");  }  } } |

# Тестирование работы

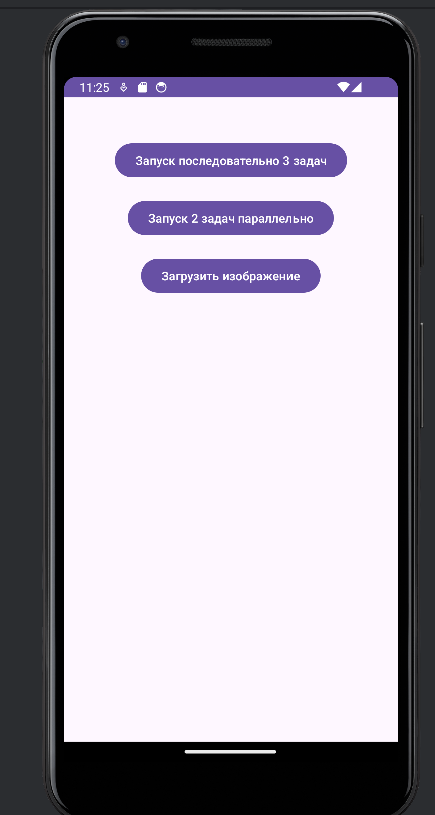


Рисунок 1. Запуск приложения

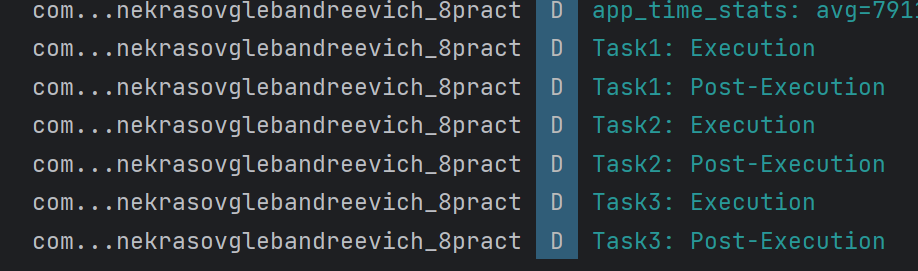


Рисунок 2. Вывод кнопки запуска последовательно 3 задач

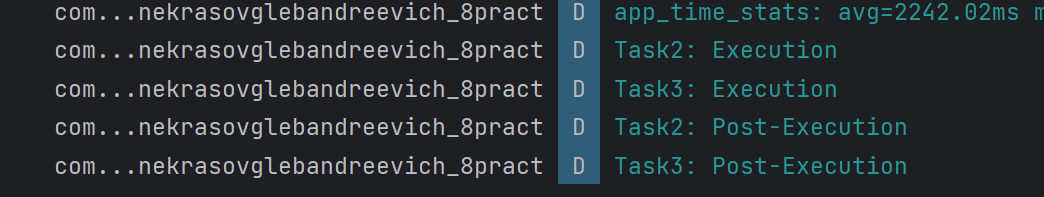


Рисунок 3. Вывод кнопки запуска параллельно 2 задач

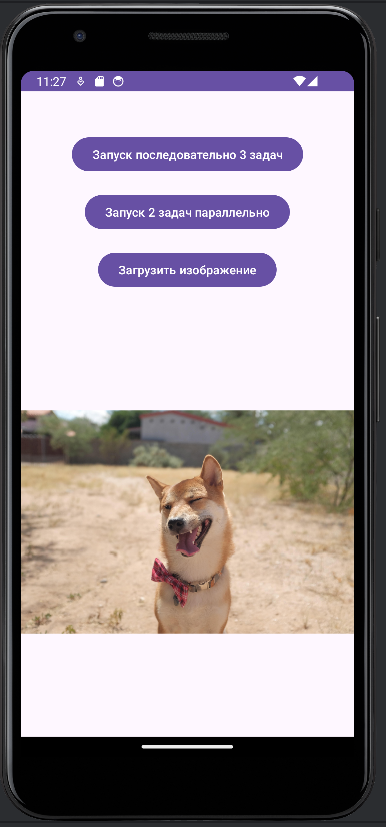


Рисунок 4. Проверка работы кнопки загрузить изображение