**LAB05 – MULTITHREADING (ANDROID)  
HƯỚNG DẪN THỰC HÀNH CHI TIẾT**

Môn học: Công Nghệ Lập Trình Đa Nền Tảng cho Ứng Dụng Di Động

Tên: PHẠM THỊ MINH PHI\_1150070033

# I. Chuẩn bị chung

- Android Studio bản mới (không bắt buộc Compose).

- Tạo project: Empty Views Activity (Java), Min SDK: API 24 trở lên, Groovy DSL.

- Tạo 5 Activity: MainActivity (Bài 1), PostActivity (Bài 2), AsyncTaskActivity (Bài 3), MusicAsyncActivity (Bài 4), MusicRxActivity (Bài 5).

- Manifest: CHỈ MainActivity là LAUNCHER (đổi tạm sang Activity khác khi muốn test trực tiếp).

- Tạo các resource dùng chung: dimens.xml và strings.xml.

- Quy tắc chung: Chỉ cập nhật UI trên UI thread (Handler/AsyncTask/RxAndroid).

## Mẫu dimens.xml

<resources>  
 <dimen name="margin\_base">5dp</dimen>  
 <dimen name="text\_small">14sp</dimen>  
 <dimen name="text\_medium">16sp</dimen>  
 <dimen name="text\_medium\_large">18sp</dimen>  
 <dimen name="text\_large">20sp</dimen>  
</resources>

## Mẫu strings.xml (bổ sung dần khi làm các bài)

<resources>  
 <string name="app\_name">Lab05\_MultiThreading</string>  
 <!-- Bài 1 -->  
 <string name="start">Start</string>  
 <string name="working">Working...</string>  
 <string name="done\_background\_thread\_has\_been\_stopped">Done\nBackground thread has been stopped</string>  
 <string name="returned\_by\_bg\_thread">Returned by background thread:\n\n</string>  
 <string name="global\_value\_seen">\nGlobal value seen by all thread</string>  
 <string name="next">Next</string>  
 <!-- Bài 2 -->  
 <string name="bg\_work\_is\_over">Background work is over!</string>  
 <string name="execute">Execute</string>  
 <string name="enter\_some\_data\_here">Enter some data here</string>  
 <!-- Bài 3 -->  
 <string name="quick\_job">Quick Job</string>  
 <string name="slow\_job">Slow Job</string>  
 <string name="please\_wait">Some SLOW job is being done. Please wait...</string>  
</resources>

# II. Bài 1 – MultiThread với Handler + Message

Mục tiêu: Tạo thread nền sinh số ngẫu nhiên và tăng biến đếm; gửi Message về UI qua Handler để cập nhật TextView/ProgressBar.

## 1) Layout – res/layout/activity\_main.xml

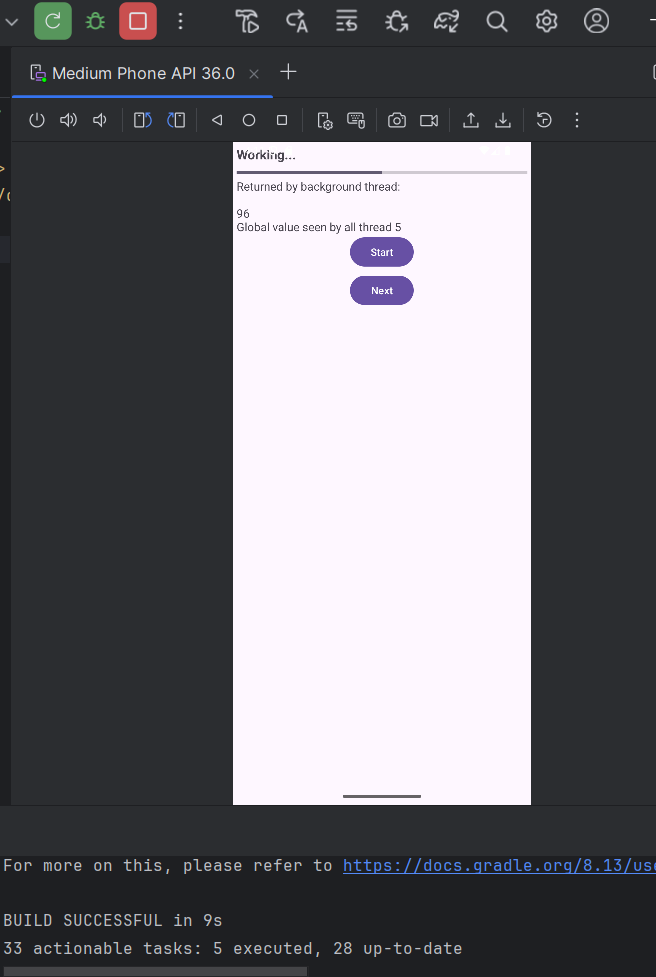
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical"  
 android:padding="@dimen/margin\_base"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
 <TextView  
 android:id="@+id/tv\_working"  
 android:textSize="@dimen/text\_medium\_large"  
 android:textStyle="bold"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"/>  
 <ProgressBar  
 android:id="@+id/pb\_first"  
 style="?android:attr/progressBarStyleHorizontal"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="@dimen/margin\_base"/>  
 <TextView  
 android:id="@+id/tv\_return"  
 android:textSize="@dimen/text\_medium"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"/>  
 <Button  
 android:id="@+id/btn\_start"  
 android:text="@string/start"  
 android:layout\_gravity="center"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"/>  
 <Button  
 android:id="@+id/btn\_next"  
 android:text="@string/next"  
 android:layout\_gravity="center"  
 android:layout\_marginTop="@dimen/margin\_base"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"/>  
</LinearLayout>

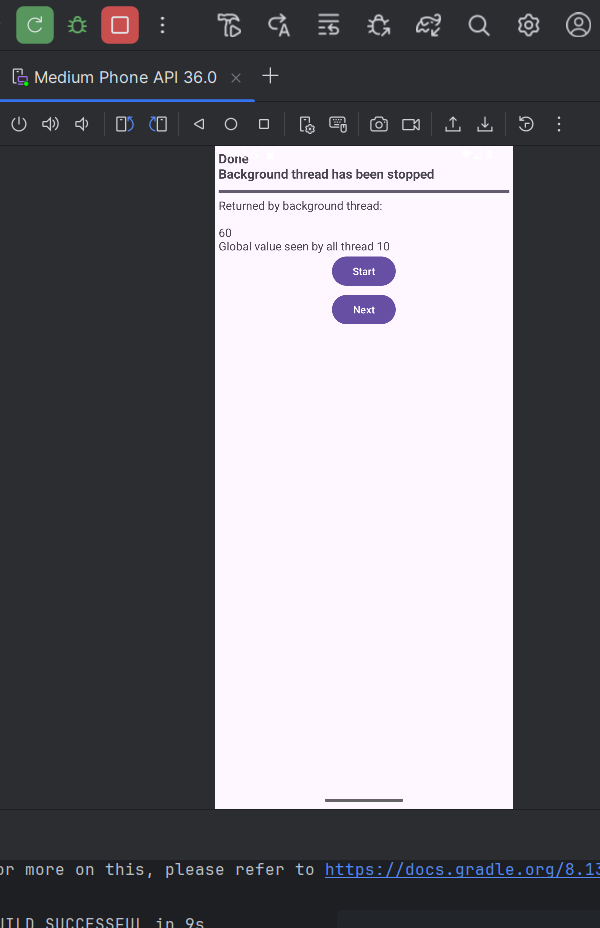
## 2) Code – MainActivity.java

package com.example.lab05\_multithreading;  
  
import androidx.appcompat.app.AppCompatActivity;  
import android.os.Bundle;  
import android.os.Handler;  
import android.os.Message;  
import android.widget.Button;  
import android.widget.ProgressBar;  
import android.widget.TextView;  
import android.content.Intent;  
import java.util.Random;  
  
public class MainActivity extends AppCompatActivity {  
 private ProgressBar pbFirst;  
 private TextView tvWorking, tvReturn;  
 private Button btnStart, btnNext;  
 private Handler handler;  
 private Thread bgThread;  
 private boolean isRunning = false;  
 private final int MAX\_SEC = 10;  
 private int intTest = 0;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
 pbFirst = findViewById(R.id.pb\_first);  
 tvWorking = findViewById(R.id.tv\_working);  
 tvReturn = findViewById(R.id.tv\_return);  
 btnStart = findViewById(R.id.btn\_start);  
 btnNext = findViewById(R.id.btn\_next);  
 pbFirst.setMax(MAX\_SEC);  
  
 handler = new Handler(msg -> {  
 String data = (String) msg.obj;  
 tvReturn.setText(data);  
 pbFirst.incrementProgressBy(1);  
 if (pbFirst.getProgress() >= MAX\_SEC) {  
 tvWorking.setText(R.string.done\_background\_thread\_has\_been\_stopped);  
 isRunning = false;  
 }  
 return true;  
 });  
  
 btnStart.setOnClickListener(v -> startBackgroundThread());  
 btnNext.setOnClickListener(v -> startActivity(new Intent(MainActivity.this, PostActivity.class)));  
 }  
  
 private void startBackgroundThread() {  
 if (isRunning) return;  
 isRunning = true;  
 tvWorking.setText(R.string.working);  
 pbFirst.setProgress(0);  
 intTest = 0;  
  
 bgThread = new Thread(() -> {  
 try {  
 Random r = new Random();  
 for (int i = 0; i < MAX\_SEC && isRunning; i++) {  
 Thread.sleep(1000);  
 intTest++;  
 String data = getString(R.string.returned\_by\_bg\_thread)  
 + r.nextInt(100)  
 + getString(R.string.global\_value\_seen) + " " + intTest;  
 Message msg = handler.obtainMessage(1, data);  
 handler.sendMessage(msg);  
 }  
 } catch (InterruptedException ignored) { }  
 });  
 bgThread.start();  
 }  
  
 @Override  
 protected void onStop() {  
 super.onStop();  
 isRunning = false;  
 }  
}

## 3) Ảnh kết quả

## 





## 4) Cách kiểm tra:

- Nhấn Start: ‘Working…’, ProgressBar chạy ~10 giây, TextView in số ngẫu nhiên + global value.

- Không crash khi rời màn hình (isRunning=false).

# III. Bài 2 – Handler.post(Runnable)

Mục tiêu: Dùng handler.post để đẩy cập nhật UI từ luồng nền thông qua Runnable; dùng synchronized bảo vệ biến chung.

## 1) Layout – res/layout/activity\_post.xml

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical"  
 android:padding="@dimen/margin\_base"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
 <TextView  
 android:id="@+id/tv\_top\_caption"  
 android:textSize="@dimen/text\_medium"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"/>  
 <ProgressBar  
 android:id="@+id/pb\_waiting"  
 style="?android:attr/progressBarStyleHorizontal"  
 android:layout\_marginTop="@dimen/margin\_base"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"/>  
 <EditText  
 android:id="@+id/et\_input"  
 android:hint="@string/enter\_some\_data\_here"  
 android:textSize="@dimen/text\_medium"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"/>  
 <Button  
 android:id="@+id/btn\_execute"  
 android:text="@string/execute"  
 android:layout\_gravity="center"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"/>  
</LinearLayout>

## 2) Code – PostActivity.java

package com.example.lab05\_multithreading;  
  
import androidx.appcompat.app.AppCompatActivity;  
import android.os.Bundle;  
import android.os.Handler;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ProgressBar;  
import android.widget.TextView;  
import android.widget.Toast;  
  
public class PostActivity extends AppCompatActivity {  
  
 private ProgressBar pbWaiting;  
 private TextView tvTopCaption;  
 private EditText etInput;  
 private Button btnExecute;  
  
 private int globalValue = 0;  
 private int accum = 0;  
  
 private Handler handler;  
 private Runnable fgRunnable, bgRunnable;  
 private Thread testThread;  
 private volatile boolean isRunning = false;  
 private static final int MAX\_PROGRESS = 20;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_post);  
  
 tvTopCaption = findViewById(R.id.tv\_top\_caption);  
 pbWaiting = findViewById(R.id.pb\_waiting);  
 etInput = findViewById(R.id.et\_input);  
 btnExecute = findViewById(R.id.btn\_execute);  
  
 initVariables();  
  
 btnExecute.setOnClickListener(v -> {  
 String input = etInput.getText().toString().trim();  
 if (input.isEmpty()) input = "(empty)";  
 Toast.makeText(this, "You said: " + input, Toast.LENGTH\_SHORT).show();  
 });  
 }  
  
 private void initVariables() {  
 handler = new Handler();  
 pbWaiting.setMax(MAX\_PROGRESS);  
  
 fgRunnable = () -> {  
 synchronized (PostActivity.this) {  
 globalValue += 100;  
 accum++;  
 }  
 tvTopCaption.setText("Global Value: " + globalValue + " | Accum: " + accum);  
 pbWaiting.setProgress(accum);  
 };  
  
 bgRunnable = () -> {  
 try {  
 for (int i = 0; i < MAX\_PROGRESS && isRunning; i++) {  
 Thread.sleep(1000);  
 synchronized (PostActivity.this) { globalValue += 1; }  
 handler.post(fgRunnable);  
 }  
 } catch (InterruptedException ignored) { }  
 };  
  
 testThread = new Thread(bgRunnable);  
 }  
  
 @Override protected void onStart() {  
 super.onStart();  
 if (!isRunning) {  
 isRunning = true;  
 testThread = new Thread(bgRunnable);  
 testThread.start();  
 }  
 }  
  
 @Override protected void onStop() {  
 super.onStop();  
 isRunning = false;  
 if (testThread != null) testThread.interrupt();  
 }  
}

## 3) Ảnh kết quả

## 

## 

## 4) Cách kiểm tra

- Thấy Global Value tăng đều; mỗi lần post lên UI giá trị nhảy +100.

- ProgressBar đạt MAX\_PROGRESS rồi dừng; Toast hiển thị khi bấm Execute.

# IV. Bài 3 – AsyncTask: Quick Job & Slow Job

Mục tiêu: Chạy tác vụ chậm ở nền bằng AsyncTask; hiển thị tiến trình qua publishProgress và ProgressDialog.

## 1) Layout – res/layout/activity\_async\_task.xml

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical" android:padding="8dp"  
 android:layout\_width="match\_parent" android:layout\_height="match\_parent">  
 <TextView  
 android:id="@+id/tv\_status"  
 android:layout\_width="match\_parent" android:layout\_height="wrap\_content"/>  
 <LinearLayout  
 android:layout\_width="match\_parent" android:layout\_height="wrap\_content"  
 android:orientation="horizontal" android:layout\_marginTop="8dp">  
 <Button  
 android:id="@+id/btn\_quick\_job"  
 android:layout\_width="0dp" android:layout\_height="wrap\_content"  
 android:layout\_weight="1" android:text="@string/quick\_job"/>  
 <Button  
 android:id="@+id/btn\_slow\_job"  
 android:layout\_width="0dp" android:layout\_height="wrap\_content"  
 android:layout\_weight="1" android:text="@string/slow\_job"  
 android:layout\_marginStart="8dp"/>  
 </LinearLayout>  
</LinearLayout>

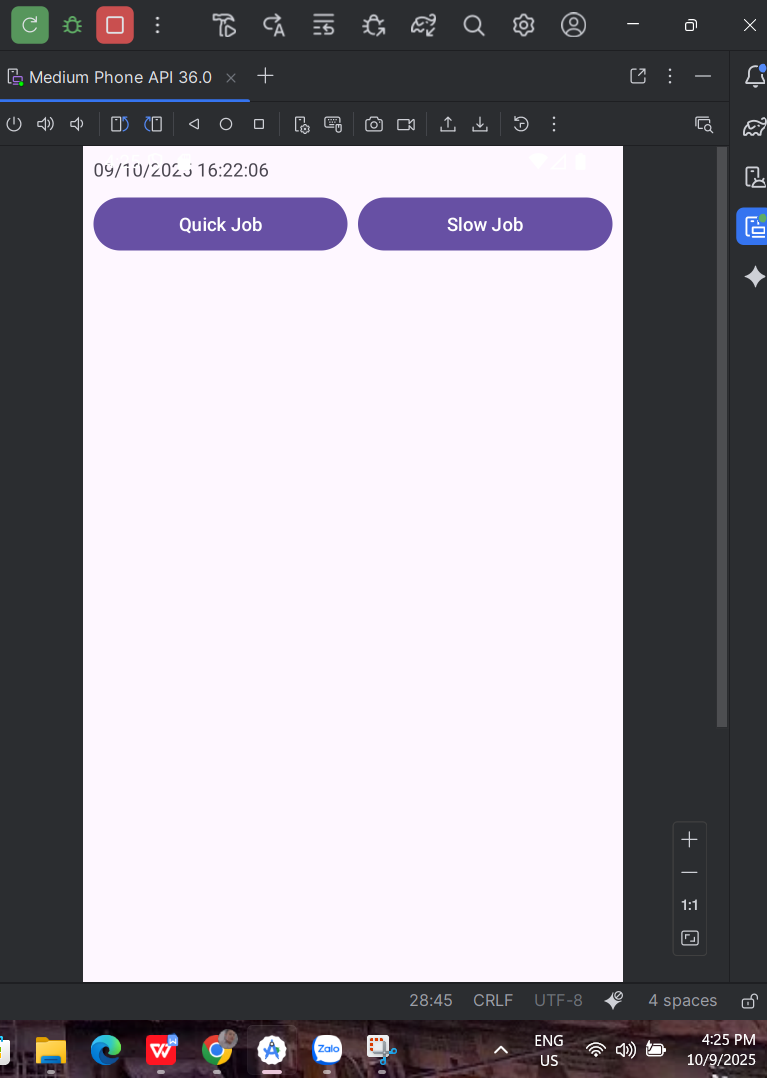
## 2) Code – AsyncTaskActivity.java

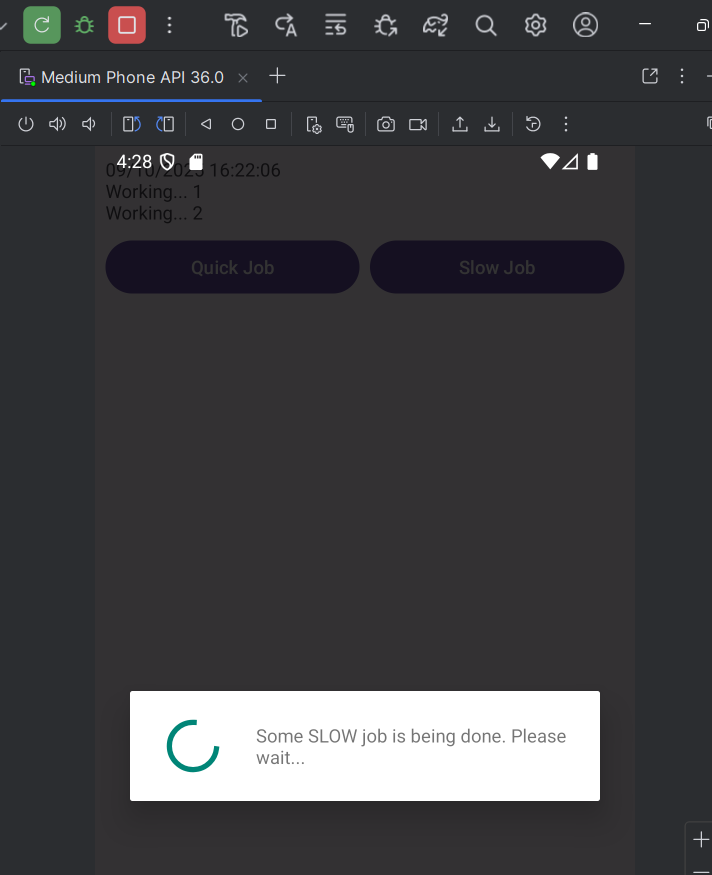
package com.example.lab05\_multithreading;  
  
import android.app.ProgressDialog;  
import android.content.Context;  
import android.os.AsyncTask;  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.TextView;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import java.text.SimpleDateFormat;  
import java.util.Date;  
import java.util.Locale;  
  
public class AsyncTaskActivity extends AppCompatActivity {  
 private Button btnQuickJob, btnSlowJob;  
 private TextView tvStatus;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_async\_task);  
  
 btnQuickJob = findViewById(R.id.btn\_quick\_job);  
 btnSlowJob = findViewById(R.id.btn\_slow\_job);  
 tvStatus = findViewById(R.id.tv\_status);  
  
 btnQuickJob.setOnClickListener(v -> {  
 String now = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss", Locale.getDefault()).format(new Date());  
 tvStatus.setText(now);  
 });  
  
 btnSlowJob.setOnClickListener(v -> new SlowTask(this, tvStatus).execute());  
 }  
  
 private static class SlowTask extends AsyncTask<String, Long, Void> {  
 private ProgressDialog pd;  
 private final Context ctx;  
 private final TextView tv;  
  
 SlowTask(Context ctx, TextView tv) { this.ctx = ctx; this.tv = tv; }  
  
 @Override protected void onPreExecute() {  
 pd = new ProgressDialog(ctx);  
 pd.setMessage(ctx.getString(R.string.please\_wait));  
 pd.setCancelable(false);  
 pd.show();  
 }  
 @Override protected Void doInBackground(String... params) {  
 try {  
 for (long i = 1; i <= 5; i++) {  
 Thread.sleep(2000);  
 publishProgress(i);  
 }  
 } catch (InterruptedException ignored) {}  
 return null;  
 }  
 @Override protected void onProgressUpdate(Long... values) {  
 tv.append("\nWorking... " + values[0]);  
 }  
 @Override protected void onPostExecute(Void result) {  
 if (pd != null && pd.isShowing()) pd.dismiss();  
 tv.append("\nDone!");  
 }  
 }  
}

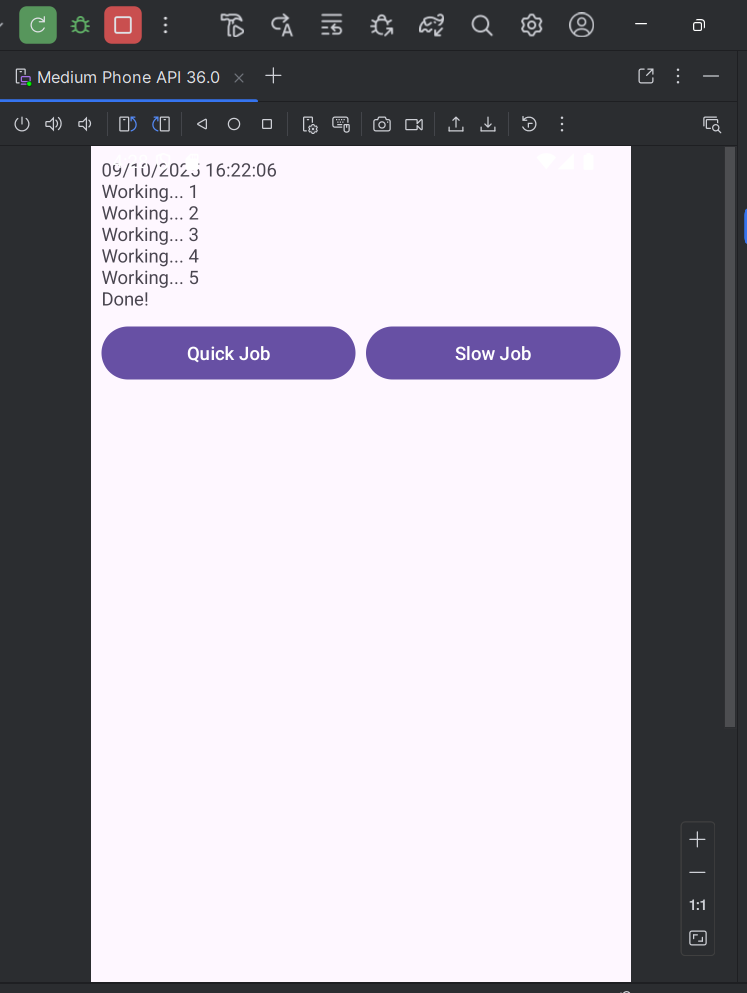
## 3) Ảnh kết quả

## 

## 







## 4) Cách kiểm tra

- Quick Job hiển thị thời gian ngay lập tức.

- Slow Job: hiện dialog ‘Please wait…’, mỗi 2s thêm ‘Working… i’, kết thúc ghi ‘Done!’.

- Không chạm UI trong doInBackground; mọi UI cập nhật ở onProgressUpdate/onPostExecute.

# V. Bài 4 – AsyncTask + MediaPlayer (phát nhạc)

Mục tiêu: Phát nhạc từ res/raw, cập nhật tiến độ bằng AsyncTask (publishProgress), có Play/Pause/Stop và SeekBar kéo để seek.

## 1) Chuẩn bị file nhạc

- Tạo thư mục res/raw/ và chép file mp3 vào (ví dụ: song.mp3 – chữ thường, không dấu, không khoảng trắng).

## 2) Layout – res/layout/activity\_music\_async.xml

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical" android:padding="12dp"  
 android:layout\_width="match\_parent" android:layout\_height="match\_parent">  
 <SeekBar  
 android:id="@+id/seekBar"  
 android:layout\_width="match\_parent" android:layout\_height="wrap\_content"/>  
 <LinearLayout  
 android:orientation="horizontal"  
 android:layout\_width="match\_parent" android:layout\_height="wrap\_content"  
 android:layout\_marginTop="12dp">  
 <Button  
 android:id="@+id/btnPlay" android:text="Play"  
 android:layout\_width="0dp" android:layout\_height="wrap\_content" android:layout\_weight="1"/>  
 <Button  
 android:id="@+id/btnPause" android:text="Pause"  
 android:layout\_width="0dp" android:layout\_height="wrap\_content" android:layout\_weight="1" android:layout\_marginStart="8dp"/>  
 <Button  
 android:id="@+id/btnStop" android:text="Stop"  
 android:layout\_width="0dp" android:layout\_height="wrap\_content" android:layout\_weight="1" android:layout\_marginStart="8dp"/>  
 </LinearLayout>  
 <TextView  
 android:id="@+id/tvTime" android:text="00:00 / 00:00"  
 android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"  
 android:layout\_marginTop="12dp"/>  
</LinearLayout>

## 3) Code – MusicAsyncActivity.java

package com.example.lab05\_multithreading;  
  
import android.media.MediaPlayer;  
import android.os.AsyncTask;  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.SeekBar;  
import android.widget.TextView;  
import androidx.appcompat.app.AppCompatActivity;  
import java.lang.ref.WeakReference;  
import java.util.Locale;  
  
public class MusicAsyncActivity extends AppCompatActivity {  
  
 private Button btnPlay, btnPause, btnStop;  
 private SeekBar seekBar;  
 private TextView tvTime;  
 private MediaPlayer mediaPlayer;  
 private ProgressTask progressTask;  
 private boolean userSeeking = false;  
  
 @Override protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_music\_async);  
 btnPlay = findViewById(R.id.btnPlay);  
 btnPause = findViewById(R.id.btnPause);  
 btnStop = findViewById(R.id.btnStop);  
 seekBar = findViewById(R.id.seekBar);  
 tvTime = findViewById(R.id.tvTime);  
 preparePlayer();  
 btnPlay.setOnClickListener(v -> startPlayback());  
 btnPause.setOnClickListener(v -> pausePlayback());  
 btnStop.setOnClickListener(v -> stopPlayback());  
 seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {  
 @Override public void onProgressChanged(SeekBar bar, int progress, boolean fromUser) {  
 if (fromUser) tvTime.setText(formatTime(progress) + " / " + formatTime(getDurationSafe()));  
 }  
 @Override public void onStartTrackingTouch(SeekBar bar) { userSeeking = true; }  
 @Override public void onStopTrackingTouch(SeekBar bar) {  
 userSeeking = false;  
 if (mediaPlayer != null) mediaPlayer.seekTo(bar.getProgress());  
 }  
 });  
 }  
  
 private void preparePlayer() {  
 mediaPlayer = MediaPlayer.create(this, R.raw.song); // đổi 'song' theo file của bạn  
 if (mediaPlayer != null) {  
 int dur = mediaPlayer.getDuration();  
 seekBar.setMax(dur);  
 tvTime.setText(formatTime(0) + " / " + formatTime(dur));  
 mediaPlayer.setOnCompletionListener(mp -> {  
 stopProgressTask();  
 seekBar.setProgress(0);  
 tvTime.setText(formatTime(0) + " / " + formatTime(getDurationSafe()));  
 });  
 }  
 }  
 private void startPlayback() {  
 if (mediaPlayer == null) preparePlayer();  
 if (mediaPlayer == null) return;  
 if (!mediaPlayer.isPlaying()) mediaPlayer.start();  
 if (progressTask == null || progressTask.getStatus() == AsyncTask.Status.FINISHED) {  
 progressTask = new ProgressTask(this, mediaPlayer);  
 progressTask.execute();  
 }  
 }  
 private void pausePlayback() {  
 if (mediaPlayer != null && mediaPlayer.isPlaying()) mediaPlayer.pause();  
 }  
 private void stopPlayback() {  
 if (mediaPlayer != null) {  
 try { mediaPlayer.stop(); } catch (IllegalStateException ignored) { }  
 releasePlayer();  
 }  
 stopProgressTask();  
 preparePlayer();  
 }  
 private void stopProgressTask() {  
 if (progressTask != null && progressTask.getStatus() != AsyncTask.Status.FINISHED) {  
 progressTask.cancel(true);  
 }  
 progressTask = null;  
 }  
 private void releasePlayer() {  
 if (mediaPlayer != null) {  
 try { mediaPlayer.reset(); mediaPlayer.release(); } catch (Exception ignored) { }  
 mediaPlayer = null;  
 }  
 }  
 private int getDurationSafe() {  
 if (mediaPlayer == null) return 0;  
 try { return mediaPlayer.getDuration(); } catch (Exception e) { return 0; }  
 }  
 private String formatTime(int ms) {  
 int totalSec = ms / 1000;  
 int m = totalSec / 60;  
 int s = totalSec % 60;  
 return String.format(Locale.getDefault(), "%02d:%02d", m, s);  
 }  
 @Override protected void onStop() {  
 super.onStop();  
 if (mediaPlayer != null && mediaPlayer.isPlaying()) mediaPlayer.pause();  
 stopProgressTask();  
 }  
 @Override protected void onDestroy() {  
 super.onDestroy();  
 stopProgressTask();  
 releasePlayer();  
 }  
  
 private static class ProgressTask extends AsyncTask<Void, Integer, Void> {  
 private final WeakReference<MusicAsyncActivity> activityRef;  
 private final WeakReference<MediaPlayer> playerRef;  
 ProgressTask(MusicAsyncActivity act, MediaPlayer mp) {  
 this.activityRef = new WeakReference<>(act);  
 this.playerRef = new WeakReference<>(mp);  
 }  
 @Override protected Void doInBackground(Void... voids) {  
 try {  
 while (!isCancelled()) {  
 MediaPlayer mp = playerRef.get();  
 MusicAsyncActivity act = activityRef.get();  
 if (mp == null || act == null) break;  
 if (mp.isPlaying()) publishProgress(mp.getCurrentPosition());  
 Thread.sleep(500);  
 }  
 } catch (InterruptedException ignored) { }  
 return null;  
 }  
 @Override protected void onProgressUpdate(Integer... values) {  
 MusicAsyncActivity act = activityRef.get();  
 MediaPlayer mp = playerRef.get();  
 if (act == null || mp == null) return;  
 int pos = values[0];  
 if (!act.userSeeking) {  
 act.seekBar.setProgress(pos);  
 act.tvTime.setText(act.formatTime(pos) + " / " + act.formatTime(act.getDurationSafe()));  
 }  
 }  
 }  
}

## 4) Ảnh kết quả

## 

## 5) Cách kiểm tra

- Play: phát nhạc, SeekBar và thời gian cập nhật ~0.5s/lần; Pause dừng, Play tiếp tục; Stop về 0.

- Kéo SeekBar để seek; rời màn hình không phát ngầm; không rò rỉ (task hủy trong onStop/onDestroy).

# VI. Bài 5 – RxJava + RxAndroid

Mục tiêu: Thay AsyncTask bằng Rx. Dùng Observable.interval để phát tick; subscribeOn(IO) và observeOn(mainThread) để cập nhật UI.

## 1) Thêm dependencies – app/build.gradle

dependencies {  
 implementation 'io.reactivex.rxjava3:rxjava:3.1.8'  
 implementation 'io.reactivex.rxjava3:rxandroid:3.0.2'  
}

## 2) Layout – có thể dùng lại activity\_music\_async.xml hoặc tạo activity\_music\_rx.xml

- Dùng lại layout của Bài 4 (SeekBar + Play/Pause/Stop + Text). ID: seekBar, btnPlay, btnPause, btnStop, tvTime.

## 3) Code – MusicRxActivity.java

package com.example.lab05\_multithreading;  
  
import android.media.MediaPlayer;  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.SeekBar;  
import android.widget.TextView;  
import androidx.appcompat.app.AppCompatActivity;  
import java.util.Locale;  
import java.util.concurrent.TimeUnit;  
import io.reactivex.rxjava3.android.schedulers.AndroidSchedulers;  
import io.reactivex.rxjava3.core.Observable;  
import io.reactivex.rxjava3.disposables.CompositeDisposable;  
import io.reactivex.rxjava3.disposables.Disposable;  
import io.reactivex.rxjava3.schedulers.Schedulers;  
  
public class MusicRxActivity extends AppCompatActivity {  
  
 private Button btnPlay, btnPause, btnStop;  
 private SeekBar seekBar;  
 private TextView tvTime;  
 private MediaPlayer mediaPlayer;  
 private final CompositeDisposable bag = new CompositeDisposable();  
 private boolean userSeeking = false;  
  
 @Override protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_music\_rx); // hoặc activity\_music\_async  
  
 btnPlay = findViewById(R.id.btnPlay);  
 btnPause = findViewById(R.id.btnPause);  
 btnStop = findViewById(R.id.btnStop);  
 seekBar = findViewById(R.id.seekBar);  
 tvTime = findViewById(R.id.tvTime);  
  
 preparePlayer();  
  
 btnPlay.setOnClickListener(v -> startPlayback());  
 btnPause.setOnClickListener(v -> pausePlayback());  
 btnStop.setOnClickListener(v -> stopPlayback());  
  
 seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {  
 @Override public void onProgressChanged(SeekBar bar, int progress, boolean fromUser) {  
 if (fromUser) tvTime.setText(fmt(progress) + " / " + fmt(getDur()));  
 }  
 @Override public void onStartTrackingTouch(SeekBar bar) { userSeeking = true; }  
 @Override public void onStopTrackingTouch(SeekBar bar) {  
 userSeeking = false;  
 if (mediaPlayer != null) mediaPlayer.seekTo(bar.getProgress());  
 }  
 });  
 }  
  
 private void preparePlayer() {  
 releasePlayer();  
 mediaPlayer = MediaPlayer.create(this, R.raw.song);  
 if (mediaPlayer == null) return;  
 seekBar.setMax(mediaPlayer.getDuration());  
 tvTime.setText(fmt(0) + " / " + fmt(getDur()));  
 mediaPlayer.setOnCompletionListener(mp -> {  
 seekBar.setProgress(0);  
 tvTime.setText(fmt(0) + " / " + fmt(getDur()));  
 bag.clear();  
 });  
 }  
  
 private void startPlayback() {  
 if (mediaPlayer == null) preparePlayer();  
 if (mediaPlayer == null) return;  
 if (!mediaPlayer.isPlaying()) mediaPlayer.start();  
  
 Disposable d = Observable.interval(0, 500, java.util.concurrent.TimeUnit.MILLISECONDS)  
 .subscribeOn(Schedulers.io())  
 .observeOn(AndroidSchedulers.mainThread())  
 .takeWhile(tick -> mediaPlayer != null)  
 .subscribe(tick -> {  
 if (mediaPlayer != null && mediaPlayer.isPlaying() && !userSeeking) {  
 int pos = mediaPlayer.getCurrentPosition();  
 seekBar.setProgress(pos);  
 tvTime.setText(fmt(pos) + " / " + fmt(getDur()));  
 }  
 }, throwable -> { /\* log nếu cần \*/ });  
  
 bag.add(d);  
 }  
  
 private void pausePlayback() {  
 if (mediaPlayer != null && mediaPlayer.isPlaying()) mediaPlayer.pause();  
 }  
 private void stopPlayback() {  
 if (mediaPlayer != null) { try { mediaPlayer.stop(); } catch (IllegalStateException ignored) {} }  
 bag.clear();  
 preparePlayer();  
 }  
 private void releasePlayer() {  
 if (mediaPlayer != null) { try { mediaPlayer.release(); } catch (Exception ignored) {} mediaPlayer = null; }  
 }  
 private int getDur() { try { return mediaPlayer != null ? mediaPlayer.getDuration() : 0; } catch (Exception e) { return 0; } }  
 private String fmt(int ms) {  
 int s = ms / 1000; int m = s / 60; s = s % 60;  
 return String.format(java.util.Locale.getDefault(), "%02d:%02d", m, s);  
 }  
 @Override protected void onStop() { super.onStop(); if (mediaPlayer != null && mediaPlayer.isPlaying()) mediaPlayer.pause(); }  
 @Override protected void onDestroy() { super.onDestroy(); bag.clear(); releasePlayer(); }  
}

## 4) Ảnh kết quả

## 

## 5) Cách kiểm tra

- Play/Pause/Stop hoạt động như Bài 4; SeekBar và thời gian cập nhật đều đặn.

- Kéo SeekBar để seek; rời màn hình không phát ngầm.

- Không rò rỉ: mọi Disposable được clear trong onDestroy().