**1. Ứng dụng MultiThread sử dụng Message**

// MainActivity.java

<?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"  
 android:orientation="vertical"  
 android:padding="@dimen/margin\_base"  
 tools:context=".MainActivity">  
  
 <TextView  
 android:id="@+id/tv\_working"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:textSize="@dimen/text\_medium\_large"  
 android:textStyle="bold" />  
  
 <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:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:textSize="@dimen/text\_medium" />  
  
 <FrameLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 <Button  
 android:id="@+id/btn\_start"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:text="@string/start" />  
  
 <ProgressBar  
 android:id="@+id/pb\_second"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:layout\_marginTop="@dimen/margin\_base" />  
  
 </FrameLayout>  
  
</LinearLayout>

Color.xml

<?xml version="1.0" encoding="utf-8"?>  
<resources>  
 <color name="colorPrimary">#2196F3</color>  
 <color name="colorPrimaryDark">#1976D2</color>  
 <color name="colorAccent">#FF4081</color>  
</resources>

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>

Strings.xml

<resources>  
 <string name="app\_name">MultiThread1</string>  
  
 <!-- Ứng dụng 1 -->  
 <string name="start">Start</string>  
 <string name="returned\_by\_bg\_thread">Returned by background thread: \n\n</string>  
 <string name="done\_background\_thread\_has\_been\_stopped">Done \nBackground thread has been stopped</string>  
 <string name="done">Done</string>  
 <string name="working">Working...</string>  
 <string name="global\_value\_seen">\n global value seen by all thread</string>  
</resources>

Style.xml

<resources>  
 <!-- Base application theme. -->  
 <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">  
 <!-- Customize your theme here. -->  
 <item name="colorPrimary">@color/colorPrimary</item>  
 <item name="colorPrimaryDark">@color/colorPrimaryDark</item>  
 <item name="colorAccent">@color/colorAccent</item>  
 </style>  
</resources>

**2. Ứng dụng MultiThread sử dụng Post**

public class PostMethodActivity extends AppCompatActivity {

private ProgressBar pbWaiting;

private TextView tvTopCaption;

private EditText etInput;

private Button btnExecute;

private int globalValue, accum;

private long startTime;

private final String PATIENCE = "Some important data is being collected now.\nPlease be patient...wait...";

private Handler handler;

private Runnable fgRunnable, bgRunnable;

private Thread testThread;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_post\_method);

findViewByIds();

initVariables();

// Xử lý execute button

btnExecute.setOnClickListener(v -> {

String input = etInput.getText().toString();

Toast.makeText(this, input, Toast.LENGTH\_SHORT).show();

});

// Start test thread

testThread = new Thread(bgRunnable);

testThread.start();

}

private void findViewByIds() {

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);

}

private void initVariables() {

globalValue = 0;

accum = 0;

startTime = System.currentTimeMillis();

handler = new Handler();

tvTopCaption.setText(PATIENCE);

fgRunnable = new Runnable() {

@Override

public void run() {

try {

// Update UI from background

accum += 100;

if (accum >= pbWaiting.getMax()) {

tvTopCaption.setText(getString(R.string.bg\_work\_is\_over));

pbWaiting.setVisibility(View.GONE);

} else {

handler.postDelayed(this, 100);

}

} catch (Exception e) {

e.printStackTrace();

}

}

};

bgRunnable = new Runnable() {

@Override

public void run() {

try {

for (int i = 0; i < 20; i++) {

Thread.sleep(1000);

synchronized (this) {

globalValue += 1;

}

handler.post(fgRunnable);

}

} catch (InterruptedException e) {

e.printStackTrace();

}

}

};

}

}

**3. Ứng dụng sử dụng AsyncTask**

public class SlowTask extends AsyncTask<String, Integer, String> {

private Context context;

private TextView tvStatus;

private ProgressDialog progressDialog;

public SlowTask(Context context, TextView tvStatus) {

this.context = context;

this.tvStatus = tvStatus;

}

@Override

protected void onPreExecute() {

super.onPreExecute();

progressDialog = ProgressDialog.show(context,

getString(R.string.please\_wait), "Slow job is running...");

}

@Override

protected String doInBackground(String... strings) {

for (int i = 0; i < 10; i++) {

try {

Thread.sleep(2000);

publishProgress(i);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

return "Job finished!";

}

@Override

protected void onProgressUpdate(Integer... values) {

super.onProgressUpdate(values);

tvStatus.setText("Working: " + values[0]);

}

@Override

protected void onPostExecute(String s) {

super.onPostExecute(s);

if (progressDialog != null && progressDialog.isShowing()) {

progressDialog.dismiss();

}

tvStatus.setText(s);

}

}

**4. Ứng dụng phát nhạc sử dụng RxJava/RxAndroid**

public class MusicPlayerActivity extends AppCompatActivity {

private static final String TAG = "MusicPlayerActivity";

private Button btnPlayMusic;

private TextView tvStatus;

private CompositeDisposable compositeDisposable;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_music\_player);

compositeDisposable = new CompositeDisposable();

btnPlayMusic = findViewById(R.id.btn\_play\_music);

tvStatus = findViewById(R.id.tv\_status);

btnPlayMusic.setOnClickListener(v -> playMusicWithRx());

}

private void playMusicWithRx() {

tvStatus.setText("Loading music...");

compositeDisposable.add(

Observable.fromCallable(this::loadAndPlayMusic)

.subscribeOn(Schedulers.io())

.observeOn(AndroidSchedulers.mainThread())

.subscribe(

result -> tvStatus.setText("Music playing: " + result),

error -> {

tvStatus.setText("Error playing music");

Log.e(TAG, "playMusicWithRx: ", error);

},

() -> tvStatus.setText("Music completed")

)

);

}

private String loadAndPlayMusic() {

// Giả lập phát nhạc

try {

for (int i = 0; i < 5; i++) {

Thread.sleep(1000);

Log.d(TAG, "Playing music segment: " + i);

}

return "Song Name - Artist";

} catch (InterruptedException e) {

throw new RuntimeException(e);

}

}

@Override

protected void onDestroy() {

super.onDestroy();

if (compositeDisposable != null) {

compositeDisposable.dispose();

}

}

}

String.xml

<resources>

<!-- Ứng dụng 1 -->

<string name="start">Start</string>

<string name="returned\_by\_bg\_thread">Returned by background thread: \n\n</string>

<string name="done\_background\_thread\_has\_been\_stopped">Done \nBackground thread has been stopped</string>

<string name="done">Done</string>

<string name="working">Working...</string>

<string name="global\_value\_seen">\n global value seen by all thread</string>

<!-- Ứng dụng 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>

<!-- Ứng dụng 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>

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>