LAB05

Bài 1:

MainActivity.java

public class MainActivity extends AppCompatActivity {  
  
 private ProgressBar pbFirst, pbSecond;  
 private TextView tvMsgWorking, tvMsgReturned;  
 private Button btnStart;  
 private boolean isRunning;  
 private int MAX\_SEC = 10;  
 private int intTest;  
 private Thread bgThread;  
 private Handler handler;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
  
 findViewByIds();  
 initVariables();  
 }  
  
 private void findViewByIds() {  
 pbFirst = findViewById(R.id.pb\_first);  
 pbSecond = findViewById(R.id.pb\_second);  
 tvMsgWorking = findViewById(R.id.tv\_working);  
 tvMsgReturned = findViewById(R.id.tv\_return);  
 btnStart = findViewById(R.id.btn\_start);  
 }  
  
 private void initVariables() {  
 isRunning = false;  
 intTest = 0;  
  
 pbFirst.setMax(MAX\_SEC);  
 pbSecond.setMax(MAX\_SEC);  
  
 // Tạo Handler để xử lý message từ background thread  
 handler = new Handler() {  
 @Override  
 public void handleMessage(Message msg) {  
 super.handleMessage(msg);  
  
 int randomNum = msg.arg1;  
 int count = msg.arg2;  
  
 pbFirst.setProgress(count);  
 tvMsgWorking.setText(getString(R.string.working));  
 tvMsgReturned.append(getString(R.string.returned\_by\_bg\_thread)  
 + "Random: " + randomNum  
 + getString(R.string.global\_value\_seen)  
 + ": " + intTest + "\n");  
  
 if (count >= MAX\_SEC) {  
 tvMsgWorking.setText(getString(R.string.done\_background\_thread\_has\_been\_stopped));  
 isRunning = false;  
 }  
 }  
 };  
  
 // Xử lý sự kiện click button Start (Bài 1)  
 btnStart.setOnClickListener(v -> {  
 if (!isRunning) {  
 isRunning = true;  
 tvMsgReturned.setText("");  
 intTest = 0;  
 pbFirst.setProgress(0);  
 initBgThread();  
 }  
 });  
  
 // Xử lý nút chuyển đổi bài tập  
 Button btnBai1 = findViewById(R.id.btn\_bai1);  
 Button btnBai2 = findViewById(R.id.btn\_bai2);  
 Button btnBai3 = findViewById(R.id.btn\_bai3);  
 Button btnBai4 = findViewById(R.id.btn\_bai4);  
  
 btnBai1.setOnClickListener(v -> {  
 // Ở lại bài 1 hoặc reload  
 startActivity(new Intent(MainActivity.this, MainActivity.class));  
 finish();  
 });  
  
 btnBai2.setOnClickListener(v -> {  
 startActivity(new Intent(MainActivity.this, Activity2.class));  
 finish();  
 });  
  
 btnBai3.setOnClickListener(v -> {  
 startActivity(new Intent(MainActivity.this, Activity3.class));  
 finish();  
 });  
  
 btnBai4.setOnClickListener(v -> {  
 startActivity(new Intent(MainActivity.this, Activity4.class));  
 finish();  
 });  
 }  
  
 private void initBgThread() {  
 bgThread = new Thread(() -> {  
 for (int i = 1; i <= MAX\_SEC && isRunning; i++) {  
 try {  
 // Tạo số ngẫu nhiên từ 0 đến 100  
 int randomNum = (int) (Math.random() \* 101);  
 intTest++;  
  
 // Tạo Message  
 Message msg = handler.obtainMessage();  
 msg.arg1 = randomNum;  
 msg.arg2 = i;  
  
 // Gửi message  
 handler.sendMessage(msg);  
  
 // Chờ 1 giây  
 Thread.sleep(1000);  
  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 });  
  
 bgThread.start();  
 }  
  
 @Override  
 protected void onStart() {  
 super.onStart();  
 }  
  
 @Override  
 protected void onStop() {  
 super.onStop();  
 isRunning = false;  
 }

DEMO:

A screenshot of a computer

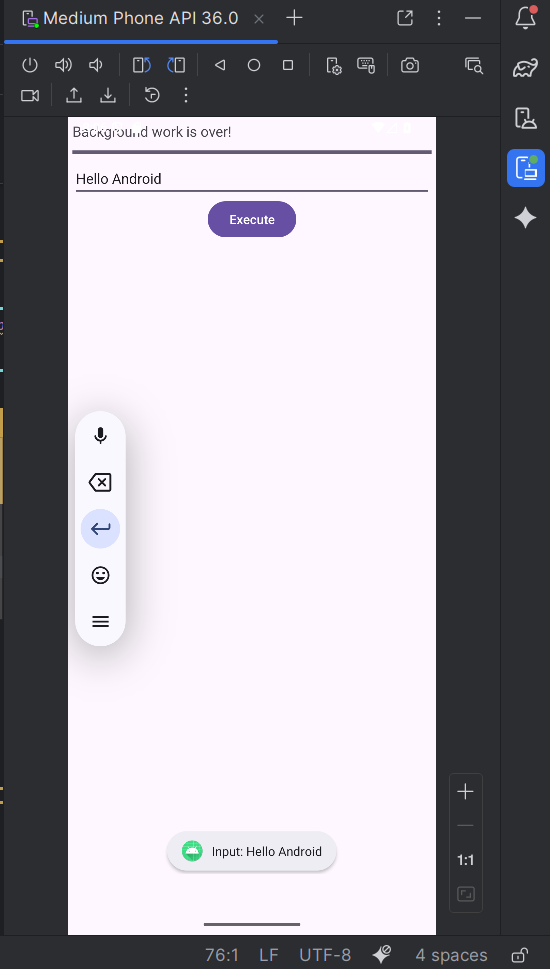
AI-generated content may be incorrect.

Bài 2:

Activity2.java

public class Activity2 extends AppCompatActivity {  
  
 private ProgressBar pbWaiting;  
 private TextView tvTopCaption;  
 private EditText etInput;  
 private Button btnExecute;  
 private int globalValue = 0;  
 private int accum = 0;  
 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\_2);  
  
 findViewByIds();  
 initVariables();  
 }  
  
 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();  
  
 // Định nghĩa bgRunnable - chạy nền, tăng globalValue mỗi 1 giây  
 bgRunnable = new Runnable() {  
 @Override  
 public void run() {  
 if (testThread != null && testThread.isAlive()) {  
 synchronized (Activity2.this) {  
 globalValue += 1;  
 }  
 tvTopCaption.setText("Global Value: " + globalValue);  
 handler.postDelayed(bgRunnable, 1000); // Chạy lại sau 1 giây  
 }  
 }  
 };  
  
 // Định nghĩa fgRunnable - chạy trước, tăng globalValue 100 mỗi lần  
 fgRunnable = new Runnable() {  
 @Override  
 public void run() {  
 if (testThread != null && testThread.isAlive()) {  
 synchronized (Activity2.this) {  
 globalValue += 100;  
 accum++;  
 }  
 pbWaiting.setProgress(accum);  
 tvTopCaption.setText("Global Value: " + globalValue);  
 handler.postDelayed(fgRunnable, 500); // Chạy lại sau 0.5 giây  
 } else {  
 // Thread kết thúc  
 tvTopCaption.setText(getString(R.string.bg\_work\_is\_over));  
 Toast.makeText(Activity2.this, "Final Global Value: " + globalValue,  
 Toast.LENGTH\_SHORT).show();  
 }  
 }  
 };  
  
 // Xử lý sự kiện Execute button  
 btnExecute.setOnClickListener(v -> {  
 String input = etInput.getText().toString();  
 if (!input.isEmpty()) {  
 Toast.makeText(Activity2.this, "Input: " + input, Toast.LENGTH\_SHORT).show();  
 }  
 });  
  
 // Khởi động background thread  
 tvTopCaption.setText(PATIENCE);  
 pbWaiting.setMax(10);  
 pbWaiting.setProgress(0);  
  
 testThread = new Thread(() -> {  
 try {  
 Thread.sleep(5000); // Chạy background task trong 5 giây  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 });  
  
 testThread.start();  
  
 // Bắt đầu postDelayed cho cả hai Runnable  
 handler.post(bgRunnable);  
 handler.post(fgRunnable);  
 }  
  
 @Override  
 protected void onDestroy() {  
 super.onDestroy();  
 // Hủy các Runnable khi Activity bị destroy  
 handler.removeCallbacks(bgRunnable);  
 handler.removeCallbacks(fgRunnable);  
 }  
}

DEMO:



Bài 3:

SlowTask.java

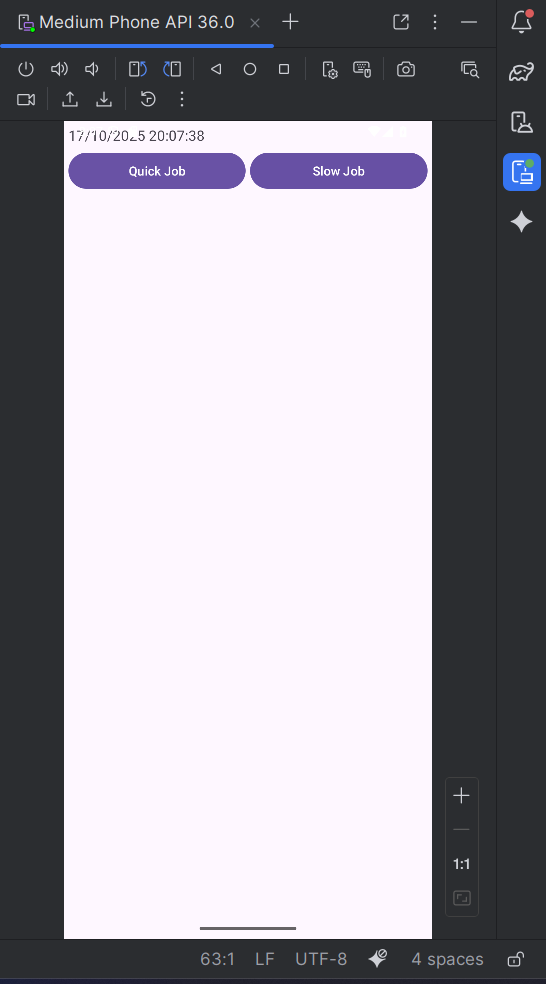
public class SlowTask extends AsyncTask<Void, Integer, String> {  
  
 private Context context;  
 private TextView tvStatus;  
 private ProgressDialog progressDialog;  
  
 public SlowTask(Context context, TextView tvStatus) {  
 this.context = context;  
 this.tvStatus = tvStatus;  
 }  
  
 // Chạy trước khi thực hiện task nền  
 @Override  
 protected void onPreExecute() {  
 super.onPreExecute();  
 progressDialog = new ProgressDialog(context);  
 progressDialog.setMessage(context.getString(R.string.please\_wait));  
 progressDialog.setProgressStyle(ProgressDialog.STYLE\_SPINNER);  
 progressDialog.setCancelable(false);  
 progressDialog.show();  
 }  
  
 // Chạy trên background thread (không được cập nhật UI trực tiếp)  
 @Override  
 protected String doInBackground(Void... voids) {  
 for (int i = 1; i <= 10; i++) {  
 try {  
 Thread.sleep(2000); // Chờ 2 giây  
 publishProgress(i); // Gửi tiến độ tới onProgressUpdate  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 return "Task Completed!";  
 }  
  
 // Cập nhật tiến độ trên UI thread  
 @Override  
 protected void onProgressUpdate(Integer... values) {  
 super.onProgressUpdate(values);  
 int progress = values[0];  
 tvStatus.setText("Progress: " + progress + "/10");  
 }  
  
 // Chạy khi task hoàn thành  
 @Override  
 protected void onPostExecute(String result) {  
 super.onPostExecute(result);  
 if (progressDialog != null && progressDialog.isShowing()) {  
 progressDialog.dismiss();  
 }  
 tvStatus.setText(result);  
 }  
}

Activity3.java

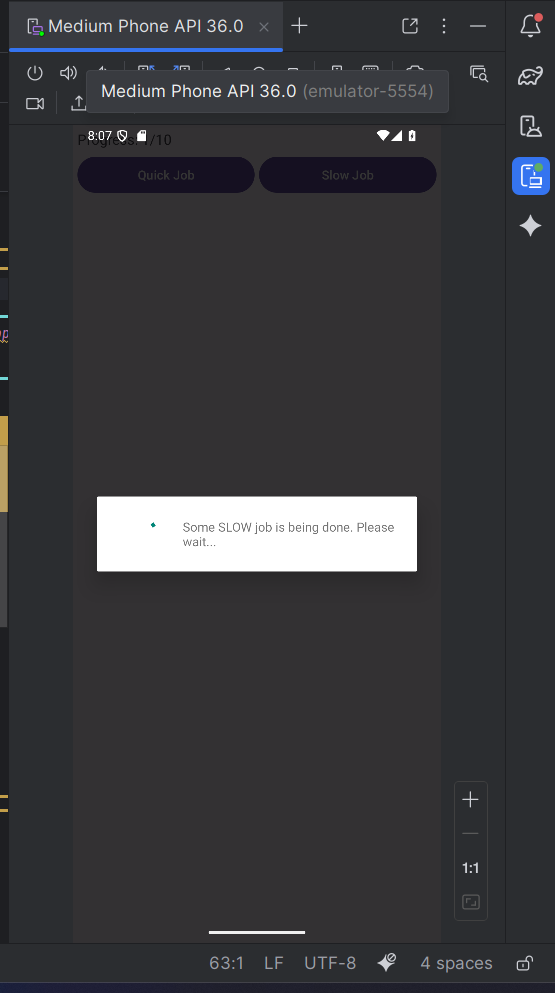
public class Activity3 extends AppCompatActivity {  
  
 private Button btnQuickJob, btnSlowJob;  
 private TextView tvStatus;  
 private SlowTask slowTask;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_3);  
  
 findViewByIds();  
 initVariables();  
 }  
  
 private void findViewByIds() {  
 btnQuickJob = findViewById(R.id.btn\_quick\_job);  
 btnSlowJob = findViewById(R.id.btn\_slow\_job);  
 tvStatus = findViewById(R.id.tv\_status);  
 }  
  
 private void initVariables() {  
 slowTask = new SlowTask(Activity3.this, tvStatus);  
  
 // Xử lý Quick Job - hiển thị thời gian hiện tại ngay lập tức  
 btnQuickJob.setOnClickListener(v -> {  
 SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");  
 tvStatus.setText(sdf.format(new Date()));  
 });  
  
 // Xử lý Slow Job - chạy AsyncTask  
 btnSlowJob.setOnClickListener(v -> {  
 // Kiểm tra nếu task chưa chạy hoặc đã hoàn thành thì mới chạy  
 if (slowTask.getStatus() == AsyncTask.Status.PENDING) {  
 slowTask.execute();  
 }  
 });  
 }

DEMO

Quick Job:



Slow Job:



Bài 4:

Activity4.java

public class Activity4 extends AppCompatActivity {  
  
 private Button btnStartRxJava;  
 private TextView tvResult;  
 private ProgressBar pbRxJava;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_4);  
  
 findViewByIds();  
 initVariables();  
 }  
  
 private void findViewByIds() {  
 btnStartRxJava = findViewById(R.id.btn\_start\_rxjava);  
 tvResult = findViewById(R.id.tv\_result);  
 pbRxJava = findViewById(R.id.pb\_rxjava);  
 }  
  
 private void initVariables() {  
 btnStartRxJava.setOnClickListener(v -> performRxJavaTask());  
 }  
  
 private void performRxJavaTask() {  
 tvResult.setText("Processing...");  
 pbRxJava.setProgress(0);  
  
 // Tạo Observable từ callable  
 Observable.fromCallable(this::doBackgroundTask)  
 // Thực hiện task trên I/O thread  
 .subscribeOn(Schedulers.io())  
 // Nhận kết quả trên main thread để cập nhật UI  
 .observeOn(AndroidSchedulers.mainThread())  
 // Subscribe với Observer  
 .subscribe(new Observer<String>() {  
 @Override  
 public void onSubscribe(Disposable d) {  
 // Gọi khi đã subscribe  
 }  
  
 @Override  
 public void onNext(String result) {  
 // Xử lý kết quả  
 tvResult.setText(result);  
 pbRxJava.setProgress(100);  
 }  
  
 @Override  
 public void onError(Throwable e) {  
 // Xử lý lỗi  
 Toast.makeText(Activity4.this, "Error: " + e.getMessage(),  
 Toast.LENGTH\_SHORT).show();  
 tvResult.setText("Error occurred!");  
 }  
  
 @Override  
 public void onComplete() {  
 // Gọi khi task hoàn thành  
 Toast.makeText(Activity4.this, "Task Completed!",  
 Toast.LENGTH\_SHORT).show();  
 }  
 });  
 }  
  
 // Phương thức chạy trên background thread  
 private String doBackgroundTask() {  
 try {  
 // Mô phỏng task mất thời gian (5 giây)  
 for (int i = 0; i < 5; i++) {  
 Thread.sleep(1000);  
 }  
 return "RxJava Task Completed Successfully!";  
 } catch (InterruptedException e) {  
 throw new RuntimeException(e);  
 }  
 }

DEMO

