SSN COLLEGE OF ENGINEERING, KALAVAKKAM

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

UCS1711 - MOBILE APPLICATION DEVELOPMENT LAB

Assignment 5

Name: Jayannthan P T　　*Dept: CSE 'A'*　　*Roll No.: 205001049*

ANDROID APPLICATION USING MULTITHREADING

Ex. No:3　　**Date:**5/9/2023

Title of the Program: Develop an android application to perform multithreading. Define 3 threads to run concurrently when "start" button is clicked.

Objective:

The objective of the Multithreading Android App project is to create an application that demonstrates the use of multithreading in Android. It defines three threads that run concurrently when the "Start" button is clicked. These threads are responsible for changing text color, moving a banner, and incrementing a counter.

Algorithm:

1. Create the main activity layout (activity_main.xml) containing TextViews for displaying various effects and buttons for controlling the threads.
2. Implement three Runnable objects, each representing a separate thread:
    - Runnable for changing text color: It changes the color of the "change_color" TextView in a loop.
    - Runnable for moving a banner: It uses ObjectAnimator to move the "moving_banner" TextView horizontally.
    - Runnable for counting: It updates the "counter" TextView and resets the count when it reaches a certain limit.
3. Create and start three threads (threadl, thread2, thread3) corresponding to the three Runnable objects.
4. Implement click event handlers for the "Start," "Resume," and "Stop" buttons:
    - "Start" button starts all three threads.
    - "Resume" button resumes the paused threads by notifying them.
    - "Stop" button stops the threads by setting a shared boolean variable (running) to false.
5. Use synchronization (lockl, lock2, lock3) to control the execution of threads and ensure proper thread communication.

6. Update UI components using runOnUiThread for UI-related actions within threads.

Features used:

- Threads: Creation and management of three concurrent threads.
- Runnable: Implementing Runnable objects for thread tasks.
- ObjectAnimator: Animating the movement of a TextView.
- Synchronization: Using locks to control thread execution and communication.
- UI updates: Updating UI components within threads using runOnUiThread.
- Button click event handling: Implementing click event handlers for buttons.

Source code:

- `MainActivity.java`

```java
package com.example.exercise6;
import androidx.appcompat.app.AppCompatActivity;
import android.animation.ObjectAnimator;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.view.animation.TranslateAnimation;
import android.widget.Button;
import android.widget.TextView;
public class MainActivity extends AppCompatActivity {
    public static Boolean running = true;
    public static Object lock1 = new Object();
    public static Object lock2 = new Object();
    public static Object lock3 = new Object();
    public int count = 0;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        TextView change_color = findViewById(R.id.change_color);
        TextView moving_banner = findViewById(R.id.moving_banner);
        TextView counter = findViewById(R.id.counter);
        Runnable runnable1 = new Runnable() {
            @Override
            public void run() {
                while (true) {
                    synchronized(lock1) {
                        if (!running) {
                            try {
                                lock1.wait();
                            } catch (InterruptedException e) {
                                e.printStackTrace();
                            }
                        }
                        runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                change_color.setTextColor(Color.parseColor("#00FF00"));
                            }
                        });
                        try {
                            Thread.sleep(500);
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        }
                        runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                change_color.setTextColor(Color.parseColor("#FFFF00"));
                            }
                        });
                        try {
                            Thread.sleep(500);
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        }
                        runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                change_color.setTextColor(Color.parseColor("#FF0000"));
                            }
                        });
                        try {
                            Thread.sleep(500);
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        }
                    }
                }
            }
        };
        Thread thread1 = new Thread(runnable1);
        Runnable runnable2 = new Runnable() {
            @Override
            public void run() {
                while (true) {
                    synchronized(lock2) {
                        if (!running) {
                            try {
                                lock2.wait();
                            } catch (InterruptedException e) {
                                e.printStackTrace();
                            }
                        }
                        runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                ObjectAnimator left_to_right =
                                    ObjectAnimator.ofFloat(moving_banner, "translationX",
-200 f, 200 f);
                                left_to_right.setDuration(2000);
                                left_to_right.start();
```

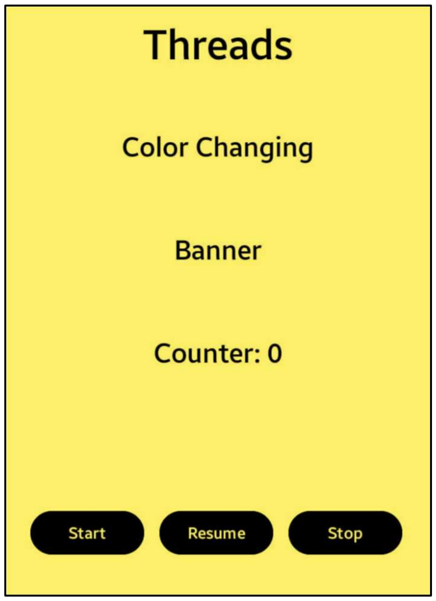
```java
                            }
                        });
                        try {
                            Thread.sleep(500);
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        }
                    }
                }
            }
        };
        Thread thread2 = new Thread(runnable2);
        Runnable runnable3 = new Runnable() {
            @Override
            public void run() {
                while (true) {
                    synchronized(lock3) {
                        if (!running) {
                            try {
                                lock3.wait();
                            } catch (InterruptedException e) {
                                e.printStackTrace();
                            }
                        }
                        runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                counter.setText("Counter: " + count);
                                count++;
                                if (count > 1000) count = 0;
                            }
                        });
                        try {
                            Thread.sleep(500);
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        }
                    }
                }
            }
        };
        Thread thread3 = new Thread(runnable3);
        Button start = findViewById(R.id.start);
        start.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                thread1.start();
                thread2.start();
                thread3.start();
            }
        });
        Button resume = findViewById(R.id.resume);
        resume.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                running = true;
                synchronized(lock1) {
                    lock1.notify();
                }
                synchronized(lock2) {
                    lock2.notify();
                }
                synchronized(lock3) {
                    lock3.notify();
                }
            }
        });
        Button stop = findViewById(R.id.stop);
        stop.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                running = false;
            }
        });
    }
}
```

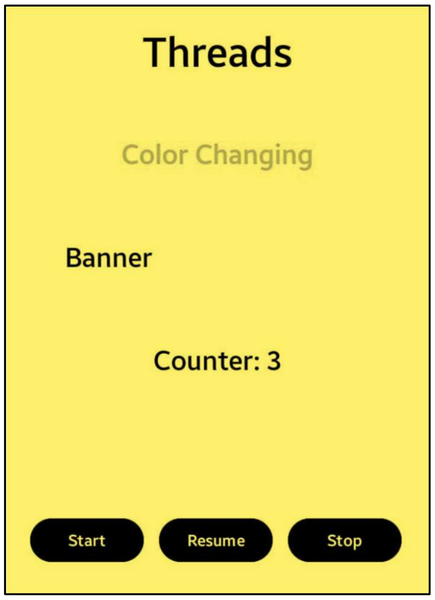
- activity_main.xml

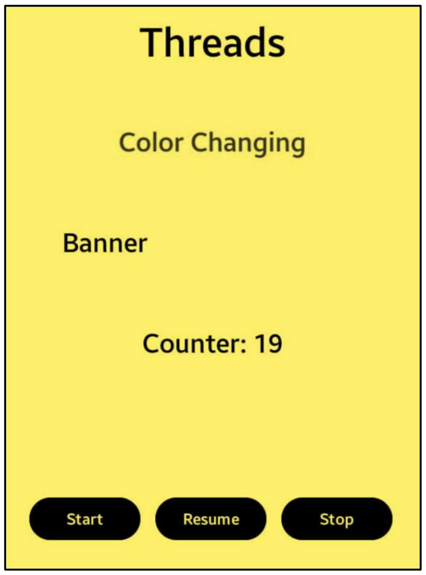
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:layout_margin="24dp"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/title"
        android:layout_width="match_parent"
        android:layout_height="100dp"
        android:gravity="center|center_vertical|center_horizontal"
        android:textSize="40dp"
        android:text="Threads"
        android:textStyle="bold"
        android:typeface="monospace"
        android:textColor="#000000" />
    <TextView android:id="@+id/change_color"
        android:layout_width="match_parent"
        android:layout_height="100dp"
        android:gravity="center|center_vertical|center_horizontal"
        android:textSize="24sp"
        android:text="Color Changing"
        android:textStyle="bold"
        android:typeface="monospace"
```

```xml
        android:textColor="#000000" />
    <TextView
        android:id="@+id/moving_banner"
        android:layout_width="match_parent"
        android:layout_height="100dp"
        android:gravity="center|center_vertical|center_horizontal"
        android:textSize="24sp"
        android:text="Banner"
        android:textStyle="bold"
        android:typeface="monospace"
        android:textColor="#000000" />
    <TextView
        android:id="@+id/counter"
        android:layout_width="match_parent"
        android:layout_height="100dp"
        android:gravity="center|center_vertical|center_horizontal"
        android:textSize="24sp"
        android:text="Counter: 0"
        android:textStyle="bold"
        android:typeface="monospace"
        android:textColor="#000000" />
    <TableLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_marginTop="100dp">
            <Button
                android:id="@+id/start"
                android:layout_width="110dp"
                android:layout_height="50dp"
                android:gravity="center|center_vertical|center_horizontal"
                android:textSize="14sp"
                android:text="Start"
                android:textStyle="bold"
                android:typeface="monospace"
                android:textColor="#ffffff"
                android:backgroundTint="#000000" />
            <Button
                android:id="@+id/resume"
                android:layout_width="110dp"
                android:layout_height="50dp"
                android:gravity="center|center_vertical|center_horizontal"
                android:layout_marginStart="15dp"
                android:textSize="14sp"
                android:text="Resume"
                android:textStyle="bold"
                android:typeface="monospace"
                android:textColor="#ffffff"
                android:backgroundTint="#000000" />
            <Button
                android:id="@+id/stop"
                android:layout_width="110dp"
                android:layout_height="50dp"
                android:gravity="center|center_vertical|center_horizontal"
                android:layout_marginStart="15dp"
                android:textSize="14sp"
                android:text="Stop"
                android:textStyle="bold"
                android:typeface="monospace"
                android:textColor="#ffffff"
                android:backgroundTint="#000000" />
        </TableRow>
    </TableLayout>
</LinearLayout>
```

Output:







Result:

The mobile application was completed successfully

Best Practices:

- Use meaningful variable and method names to improve code readability.
- Employ synchronization mechanisms to prevent race conditions and ensure proper thread
- communication.
- Implement proper error handling and exception catching for robust code.
- Separate UI-related code from background thread code using runOnUIThread.
- Follow Android coding conventions and design guidelines for a consistent and user-friendly
- app.
- Consider using constants or resources for hard-coded values like color codes and duration.

Learning Outcomes:

- Demonstrating multithreading in an Android application.
- Creating and managing threads to perform concurrent tasks.
- Implementing Runnable objects to define the behavior of threads.
- Using ObjectAnimator for animations within Android apps.
- Utilizing synchronization to control thread execution and communication.
- Handling button click events for user interaction.
- Gaining insights into proper multithreading practices and error handling in Android
- development.