**INSTITUTO TECNOLOGICO ESCUELA SUPERIOR PEDRO DOMIGO MURILLO**



**INFORME DE LABORATORIO N°7**

**PSM-400 PROGRAMACION DE SISTEMAS MOVILES**

**ESTUDIANTE: Angela Mamani Ayca**

**DOCENTE: Lic. Erik V. Huallpa A.**

**LA PAZ – BOLIVIA**

*Informe de Proyecto Android – Escáner QR y Aplicaciones Auxiliares*

***1. Introducción***

*En el presente proyecto se desarrollaron varias aplicaciones móviles en Android con el fin de reforzar los conocimientos adquiridos sobre el uso de interfaces, manipulación de hardware y análisis de información en tiempo real.*

*Los ejercicios incluyen: una calculadora científica, un conversor de unidades, una linterna utilizando el flash del dispositivo y un escáner de códigos QR.*

*Cada aplicación tiene como finalidad mejorar la comprensión de la programación en Java, el manejo de componentes de Android y la integración de librerías externas como ML Kit.*

***2. Marco Teórico***

*Android Studio: Entorno de desarrollo oficial para crear aplicaciones Android utilizando Java o Kotlin.*

*CameraX: API de Android que permite integrar fácilmente la cámara en aplicaciones con control sobre la vista previa, análisis de imágenes y compatibilidad con distintos dispositivos.*

*ML Kit Barcode Scanning: Librería de Google que permite detectar y leer códigos de barras y QR en tiempo real.*

*Componentes de UI: EditText, TextView, Button, Spinner y PreviewView son esenciales para la interacción con el usuario.*

*Permisos: La seguridad de Android requiere solicitar permisos explícitos para acceder al hardware, como la cámara.*

***3. Objetivo General***

*Desarrollar aplicaciones móviles en Android que permitan realizar cálculos, conversiones, control de hardware y detección de códigos QR, aplicando los conocimientos adquiridos en programación, diseño de interfaces y manejo de librerías externas.*

***4. Objetivos Específicos***

*Implementar una calculadora científica que permita operaciones básicas y funciones avanzadas, manteniendo cálculos en memoria.*

*Crear un conversor de unidades para longitud, peso y temperatura utilizando fórmulas matemáticas directas.*

*Desarrollar una linterna que utilice el flash del dispositivo con opción de parpadeo intermitente.*

*Integrar un escáner de códigos QR que detecte y muestre información en tiempo real usando CameraX y ML Kit.*

***5. Desarrollo de los Ejercicios***

***Ejercicio 1: Calculadora Científica***

*Se desarrolló una interfaz sencilla con botones para operaciones básicas y científicas.*

*Se utilizó un EditText para ingresar números y un TextView para mostrar resultados.*

*Los cálculos se almacenan temporalmente para permitir operaciones consecutivas sin perder el valor anterior.*

***Ejercicio 2: Conversor de Unidades***

*Se implementaron conversiones de longitud (metros, kilómetros, pulgadas), peso (kilogramos, libras) y temperatura (Celsius, Fahrenheit, Kelvin).*

*La selección de unidades se realizó mediante Spinner y los resultados se mostraron en TextView.*

*Se emplearon fórmulas matemáticas directas para cada conversión.*

***Ejercicio 3: Linterna Multinacional***

*Se accedió al flash de la cámara mediante Camera API.*

*Se desarrolló una función de encendido/apagado con un Button.*

*Se implementó un parpadeo intermitente controlado por Handler para simular señalizaciones luminosas.*

***Ejercicio 4: Escáner de Códigos QR***

*Se utilizó CameraX para mostrar la vista previa en un PreviewView.*

*Se configuró ML Kit Barcode Scanning para detectar códigos QR y otros códigos de barra.*

*Los frames de la cámara se analizaron en segundo plano mediante ExecutorService.*

*El contenido de cada QR detectado se mostró en un TextView en tiempo real, sin almacenarse.*

***Flujo de QR Scanner:***

*La app solicita permiso de cámara.*

*Se abre la cámara trasera y se muestra la vista previa.*

*Cada frame se analiza en tiempo real.*

*Si se detecta un QR, se muestra su contenido en pantalla.*

*La cámara continúa activa hasta que se cierra la aplicación.*

***6. Conclusiones***

*Se logró desarrollar cuatro aplicaciones funcionales en Android que combinan cálculo, conversión, control de hardware y visión por computadora.*

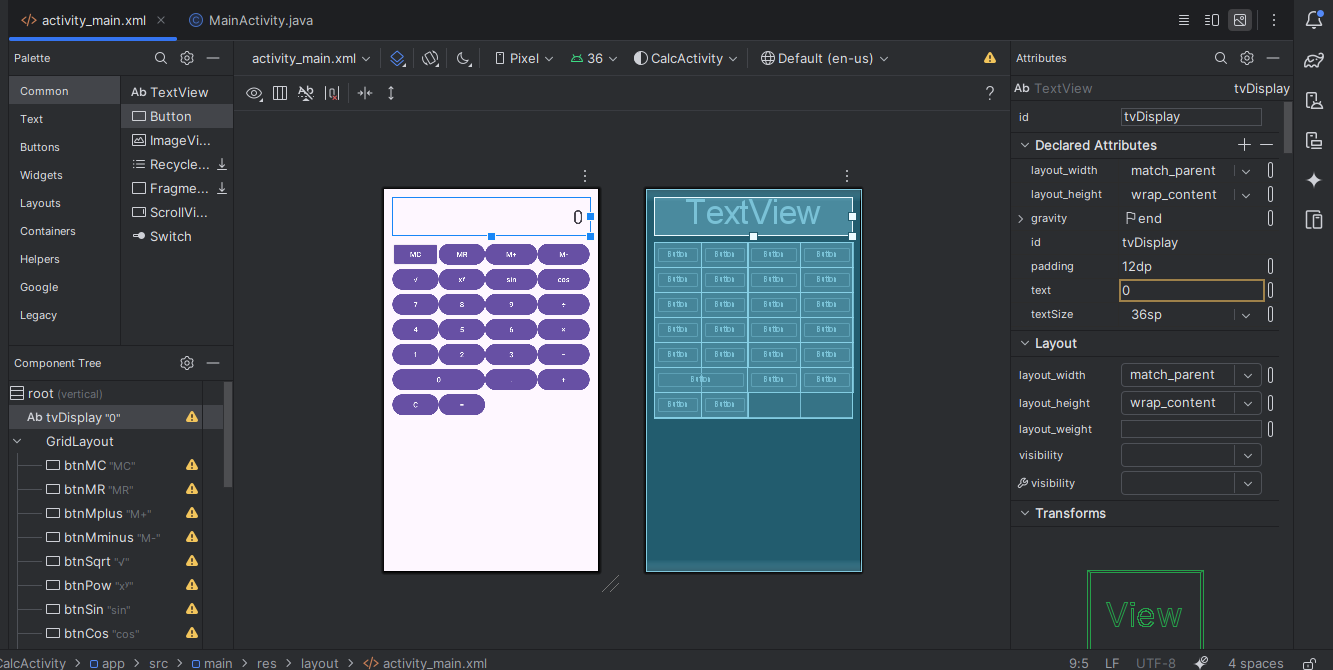
*La integración de CameraX y ML Kit permite analizar imágenes en tiempo real de manera eficiente.*

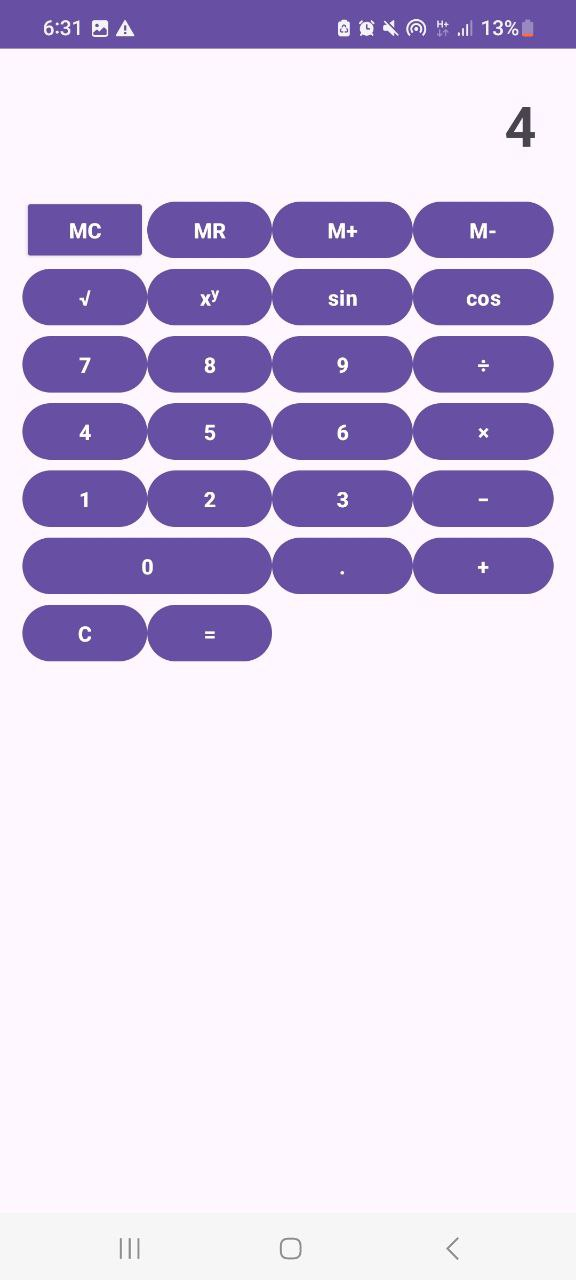
*El proyecto refuerza el aprendizaje sobre interfaces de usuario, permisos, ejecución en segundo plano y uso de librerías externas.*

*Cada ejercicio demuestra la importancia de la programación orientada a objetos y la modularidad en el desarrollo de aplicaciones móviles.*

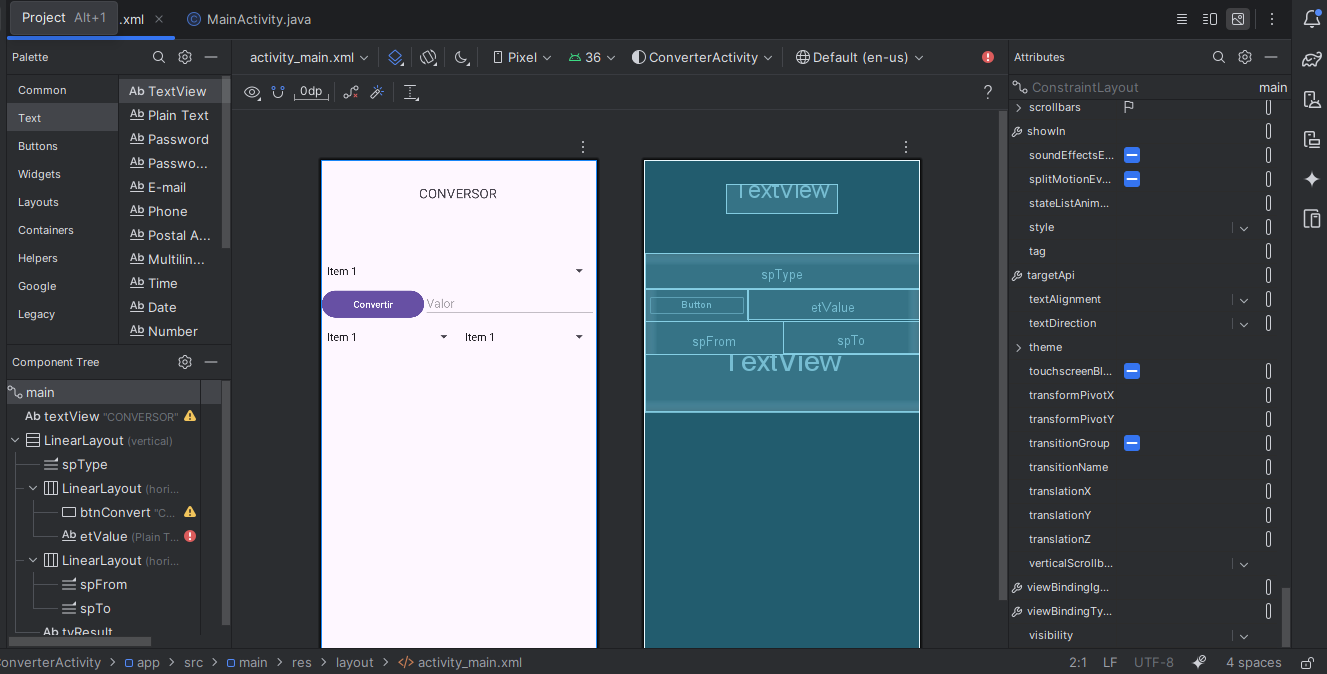
*4 EJERCICIO PROPUESTOS*

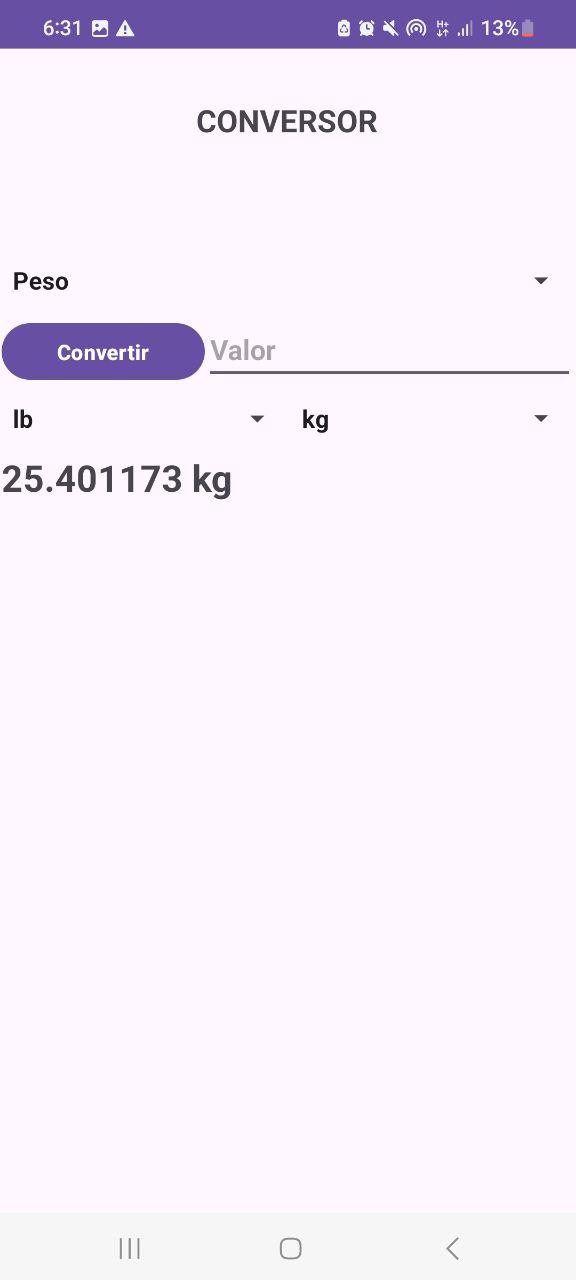
package com.angela.calcactivity;  
  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.TextView;  
import androidx.appcompat.app.AppCompatActivity;  
import java.util.Locale;  
  
public class MainActivity extends AppCompatActivity implements View.OnClickListener {  
 TextView tv;  
 double memory = 0.0;  
 String current = "";  
 double operand = 0;  
 String operator = "";  
 boolean isNew = true;  
  
 @Override protected void onCreate(Bundle s){  
 super.onCreate(s);  
 setContentView(R.layout.*activity\_main*);  
 tv = findViewById(R.id.*tvDisplay*);  
  
 int[] ids = {R.id.*btn0*,R.id.*btn1*,R.id.*btn2*,R.id.*btn3*,R.id.*btn4*,R.id.*btn5*,R.id.*btn6*,R.id.*btn7*,R.id.*btn8*,R.id.*btn9*,R.id.*btnDot*,  
 R.id.*btnAdd*,R.id.*btnSub*,R.id.*btnMul*,R.id.*btnDiv*,R.id.*btnEqual*,R.id.*btnClear*,  
 R.id.*btnMC*,R.id.*btnMR*,R.id.*btnMplus*,R.id.*btnMminus*,  
 R.id.*btnSqrt*,R.id.*btnPow*,R.id.*btnSin*,R.id.*btnCos*};  
 for(int id: ids) findViewById(id).setOnClickListener(this);  
 }  
  
 @Override public void onClick(View v){  
 int id = v.getId();  
 if(id==R.id.*btnClear*){ current=""; operator=""; operand=0; isNew=true; tv.setText("0"); return; }  
 if(id==R.id.*btnDot*){ if(!current.contains(".")) current += current.isEmpty()?"0.":"."; tv.setText(current); return; }  
 if(id>=R.id.*btn0* && id<=R.id.*btn9*){  
 String d = ((Button)v).getText().toString();  
 current = isNew ? d : current + d;  
 isNew = false;  
 tv.setText(current);  
 return;  
 }  
 if(id==R.id.*btnAdd* || id==R.id.*btnSub* || id==R.id.*btnMul* || id==R.id.*btnDiv*){  
 commitPending();  
 operator = ((Button)v).getText().toString();  
 isNew = true;  
 return;  
 }  
 if(id==R.id.*btnEqual*){  
 commitPending();  
 operator = "";  
 isNew = true;  
 tv.setText(format(operand));  
 current = tv.getText().toString();  
 return;  
 }  
 if(id==R.id.*btnMC*){ memory = 0; return; }  
 if(id==R.id.*btnMR*){ tv.setText(format(memory)); current = format(memory); isNew = true; return; }  
 if(id==R.id.*btnMplus*){ memory += parseCurrent(); return; }  
 if(id==R.id.*btnMminus*){ memory -= parseCurrent(); return; }  
  
 if(id==R.id.*btnSqrt*){ double v1 = Math.*sqrt*(parseCurrent()); tv.setText(format(v1)); current = format(v1); isNew=true; return; }  
 if(id==R.id.*btnPow*){  
 commitPending(); operator="^"; isNew=true; return;  
 }  
 if(id==R.id.*btnSin*){ double v2 = Math.*sin*(Math.*toRadians*(parseCurrent())); tv.setText(format(v2)); current = format(v2); isNew=true; return; }  
 if(id==R.id.*btnCos*){ double v3 = Math.*cos*(Math.*toRadians*(parseCurrent())); tv.setText(format(v3)); current = format(v3); isNew=true; return; }  
 }  
  
 private double parseCurrent(){  
 if(current==null || current.isEmpty()) return 0;  
 try{ return Double.*parseDouble*(current); } catch(Exception e){ return 0; }  
 }  
  
 private void commitPending(){  
 double cur = parseCurrent();  
 if(operator==null || operator.isEmpty()){  
 operand = cur;  
 } else {  
 switch(operator){  
 case "+": operand = operand + cur; break;  
 case "−":   
 case "-": operand = operand - cur; break;  
 case "×": operand = operand \* cur; break;  
 case "÷": operand = operand / (cur==0?1:cur); break;  
 case "^": operand = Math.*pow*(operand, cur); break;  
 }  
 }  
 tv.setText(format(operand));  
 current = format(operand);  
 }  
  
 private String format(double v){  
 if(Math.*round*(v) == v) return String.*format*(Locale.*US*,"%.0f", v);  
 return String.*format*(Locale.*US*,"%.8f", v).replaceAll("0+$","").replaceAll("\\.$","");  
 }  
}



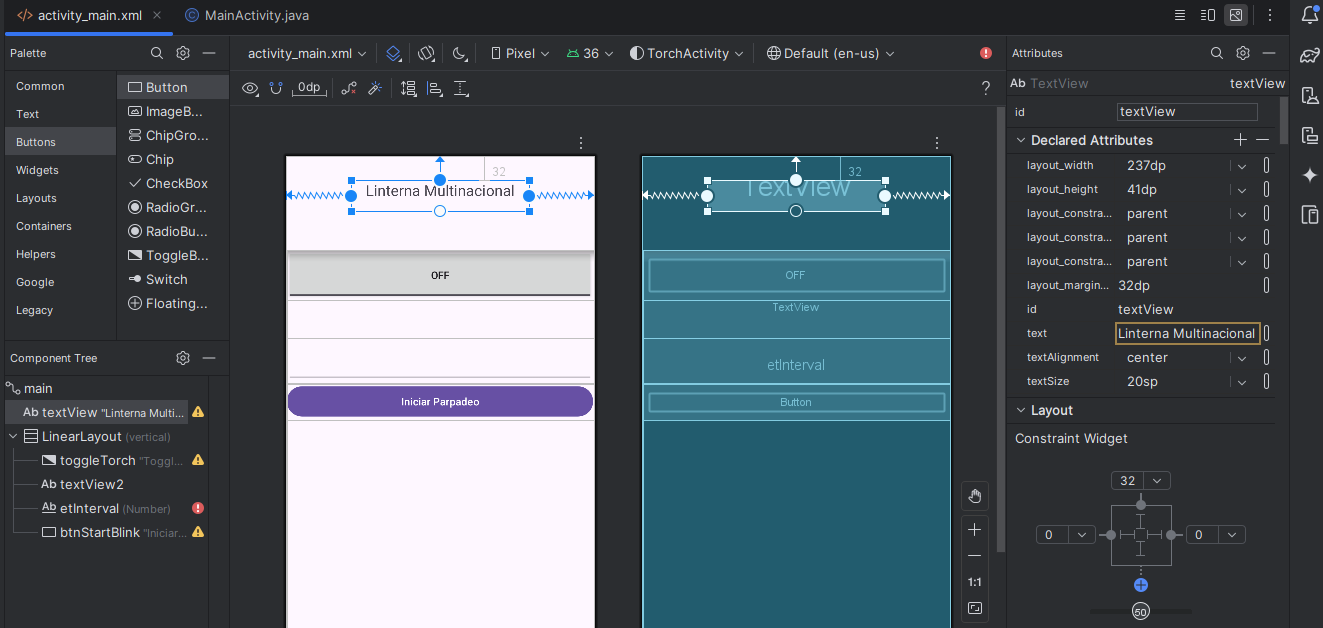


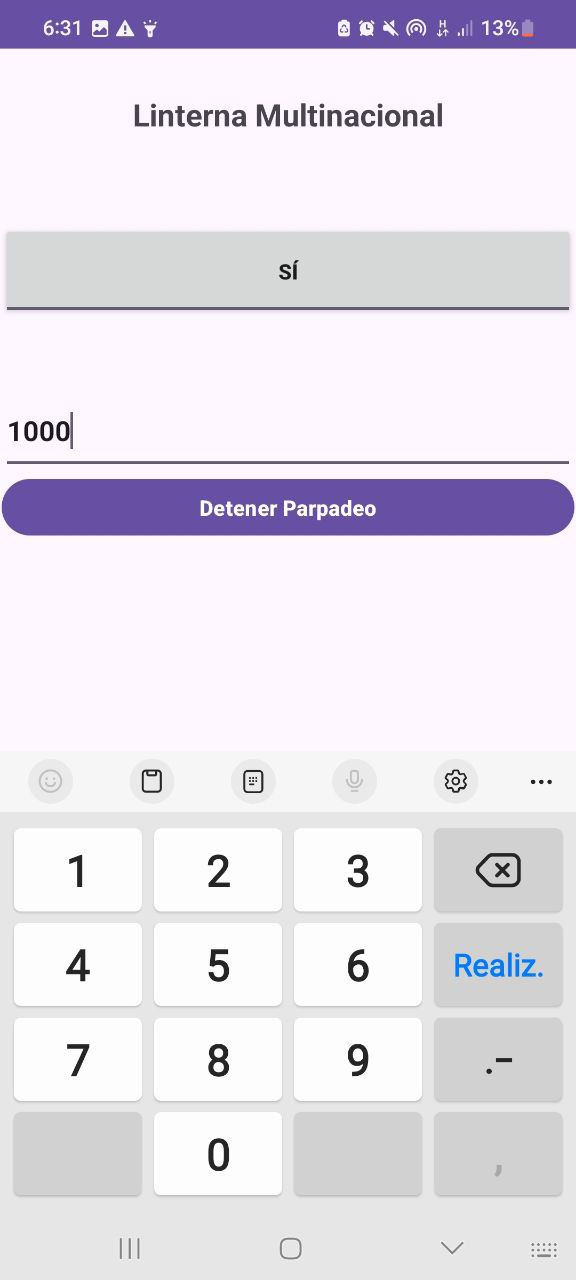
package com.angela.converteractivity;  
  
  
import android.os.Bundle;  
import android.view.View;  
import android.widget.\*;  
import androidx.appcompat.app.AppCompatActivity;  
import java.util.Arrays;  
  
public class MainActivity extends AppCompatActivity {  
 Spinner spType, spFrom, spTo;  
 EditText etValue;  
 TextView tvResult;  
 Button btnConvert;  
  
 String[] types = {"Longitud","Peso","Temperatura"};  
 String[] len = {"m","km","cm","mm"};  
 String[] weight = {"kg","g","lb"};  
 String[] temp = {"C","F","K"};  
  
 @Override protected void onCreate(Bundle s){  
 super.onCreate(s);  
 setContentView(R.layout.*activity\_main*);  
 spType = findViewById(R.id.*spType*);  
 spFrom = findViewById(R.id.*spFrom*);  
 spTo = findViewById(R.id.*spTo*);  
 etValue = findViewById(R.id.*etValue*);  
 tvResult = findViewById(R.id.*tvResult*);  
 btnConvert = findViewById(R.id.*btnConvert*);  
  
 ArrayAdapter<String> adTypes = new ArrayAdapter<>(this, android.R.layout.*simple\_spinner\_item*, types);  
 adTypes.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);  
 spType.setAdapter(adTypes);  
  
 spType.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener(){  
 @Override public void onItemSelected(AdapterView<?> p, View v, int pos, long id){  
 String[] items = pos==0?len: pos==1?weight: temp;  
 ArrayAdapter<String> ad = new ArrayAdapter<>(MainActivity.this, android.R.layout.*simple\_spinner\_item*, items);  
 ad.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);  
 spFrom.setAdapter(ad); spTo.setAdapter(ad);  
 }  
 @Override public void onNothingSelected(AdapterView<?> p){}  
 });  
  
 btnConvert.setOnClickListener(view -> {  
 String from = (String)spFrom.getSelectedItem();  
 String to = (String)spTo.getSelectedItem();  
 double v = 0;  
 try{ v = Double.*parseDouble*(etValue.getText().toString()); } catch(Exception e){ tvResult.setText("Ingresa un número"); return; }  
 double res = 0;  
 switch((String)spType.getSelectedItem()){  
 case "Longitud": res = convertLength(v, from, to); break;  
 case "Peso": res = convertWeight(v, from, to); break;  
 case "Temperatura": res = convertTemp(v, from, to); break;  
 }  
 tvResult.setText(String.*format*("%.6f %s", res, to));  
 });  
 }  
  
 double convertLength(double val, String from, String to){  
 double meters = val \* (from.equals("km")?1000: from.equals("m")?1: from.equals("cm")?0.01:0.001);  
 double toFactor = to.equals("km")?0.001: to.equals("m")?1: to.equals("cm")?100:1000;  
 return meters \* toFactor;  
 }  
  
 double convertWeight(double val, String from, String to){  
  
 double kg = val \* (from.equals("kg")?1: from.equals("g")?0.001: 0.45359237);  
 double toFactor = to.equals("kg")?1: to.equals("g")?1000: 1/0.45359237;  
 return kg \* toFactor;  
 }  
 double convertTemp(double val, String from, String to){  
 double c = 0;  
 if(from.equals("C")) c = val;  
 if(from.equals("F")) c = (val - 32) \* 5/9;  
 if(from.equals("K")) c = val - 273.15;  
 if(to.equals("C")) return c;  
 if(to.equals("F")) return c \* 9/5 + 32;  
 return c + 273.15;  
 }  
}





package com.angela.torchactivity;  
  
  
import android.Manifest;  
import android.content.pm.PackageManager;  
import android.hardware.camera2.CameraAccessException;  
import android.hardware.camera2.CameraManager;  
import android.os.Bundle;  
import android.os.Handler;  
import android.view.View;  
import android.widget.\*;  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.app.ActivityCompat;  
  
public class MainActivity extends AppCompatActivity {  
 private CameraManager cameraManager;  
 private String cameraId;  
 ToggleButton toggle;  
 Button btnBlink;  
 EditText etInterval;  
 Handler handler = new Handler();  
 boolean blinking = false;  
 Runnable blinkRunnable;  
  
 final int REQ = 100;  
  
 @Override protected void onCreate(Bundle s){  
 super.onCreate(s);  
 setContentView(R.layout.*activity\_main*);  
 toggle = findViewById(R.id.*toggleTorch*);  
 btnBlink = findViewById(R.id.*btnStartBlink*);  
 etInterval = findViewById(R.id.*etInterval*);  
  
 if (ActivityCompat.*checkSelfPermission*(this, Manifest.permission.*CAMERA*) != PackageManager.*PERMISSION\_GRANTED*) {  
 ActivityCompat.*requestPermissions*(this, new String[]{Manifest.permission.*CAMERA*}, REQ);  
 }  
  
 cameraManager = (CameraManager)getSystemService(*CAMERA\_SERVICE*);  
 try{  
 for(String id : cameraManager.getCameraIdList()){  
   
 if(cameraManager.getCameraCharacteristics(id).get(android.hardware.camera2.CameraCharacteristics.*FLASH\_INFO\_AVAILABLE*)){  
 cameraId = id; break;  
 }  
 }  
 } catch (CameraAccessException e){ e.printStackTrace(); }  
  
 toggle.setOnCheckedChangeListener((buttonView, isChecked) -> {  
 setTorch(isChecked);  
 });  
  
 btnBlink.setOnClickListener(v -> {  
 if(blinking){  
 stopBlink();  
 } else {  
 String sInterval = etInterval.getText().toString();  
 int interval = 500;  
 try{ interval = Integer.*parseInt*(sInterval); }catch(Exception e){}  
 startBlink(interval);  
 }  
 });  
 }  
  
 void setTorch(boolean on){  
 if(cameraId==null) return;  
 try{ cameraManager.setTorchMode(cameraId, on); } catch(CameraAccessException e){ e.printStackTrace(); }  
 }  
  
 void startBlink(int intervalMs){  
 blinking = true;  
 btnBlink.setText("Detener Parpadeo");  
 blinkRunnable = new Runnable(){  
 boolean state = false;  
 @Override public void run(){  
 state = !state;  
 setTorch(state);  
 toggle.setChecked(state);  
 handler.postDelayed(this, intervalMs);  
 }  
 };  
 handler.post(blinkRunnable);  
 }  
  
 void stopBlink(){  
 blinking = false;  
 btnBlink.setText("Iniciar Parpadeo");  
 if(blinkRunnable!=null) handler.removeCallbacks(blinkRunnable);  
 setTorch(false);  
 toggle.setChecked(false);  
 }  
  
 @Override protected void onPause(){  
 super.onPause();  
 stopBlink();  
 }  
  
 @Override public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults){  
 if(requestCode==REQ){  
 if(grantResults.length==0 || grantResults[0]!=PackageManager.*PERMISSION\_GRANTED*){  
 Toast.*makeText*(this,"Permiso de cámara requerido para usar el flash", Toast.*LENGTH\_SHORT*).show();  
 }  
 } else super.onRequestPermissionsResult(requestCode, permissions, grantResults);  
 }  
}





package com.angela.qrscanneractivity;  
  
import android.Manifest;  
import android.content.pm.PackageManager;  
import android.os.Bundle;  
import android.util.Size;  
import android.widget.TextView;  
import android.widget.Toast;  
import androidx.annotation.NonNull;  
import androidx.annotation.OptIn;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.camera.core.\*;  
import androidx.camera.lifecycle.ProcessCameraProvider;  
import androidx.camera.view.PreviewView;  
import androidx.core.app.ActivityCompat;  
import androidx.core.content.ContextCompat;  
import com.google.common.util.concurrent.ListenableFuture;  
import com.google.mlkit.vision.barcode.BarcodeScanner;  
import com.google.mlkit.vision.barcode.BarcodeScannerOptions;  
import com.google.mlkit.vision.barcode.BarcodeScanning;  
import com.google.mlkit.vision.common.InputImage;  
import java.util.concurrent.ExecutionException;  
import java.util.concurrent.ExecutorService;  
import java.util.concurrent.Executors;  
import com.google.mlkit.vision.barcode.common.Barcode;  
  
public class MainActivity extends AppCompatActivity {  
 private PreviewView previewView;  
 private TextView tvInfo;  
 private ExecutorService cameraExecutor;  
 private final int REQ\_CODE = 200;  
  
 @Override protected void onCreate(Bundle s){  
 super.onCreate(s);  
 setContentView(R.layout.*activity\_main*);  
 previewView = findViewById(R.id.*previewView*);  
 tvInfo = findViewById(R.id.*tvInfo*);  
 cameraExecutor = Executors.*newSingleThreadExecutor*();  
  
 if(ContextCompat.*checkSelfPermission*(this, Manifest.permission.*CAMERA*) == PackageManager.*PERMISSION\_GRANTED*){  
 startCamera();  
 } else {  
 ActivityCompat.*requestPermissions*(this, new String[]{Manifest.permission.*CAMERA*}, REQ\_CODE);  
 }  
 }  
  
 private void startCamera(){  
 ListenableFuture<ProcessCameraProvider> cameraProviderFuture = ProcessCameraProvider.*getInstance*(this);  
 cameraProviderFuture.addListener(() -> {  
 try{  
 ProcessCameraProvider cameraProvider = cameraProviderFuture.get();  
 bindPreviewAndAnalyzer(cameraProvider);  
 } catch (ExecutionException | InterruptedException e){ e.printStackTrace(); }  
 }, ContextCompat.*getMainExecutor*(this));  
 }  
  
 @OptIn(markerClass = ExperimentalGetImage.class)  
 private void bindPreviewAndAnalyzer(ProcessCameraProvider cameraProvider){  
 cameraProvider.unbindAll();  
  
 Preview preview = new Preview.Builder()  
 .setTargetResolution(new Size(1280,720))  
 .build();  
  
 CameraSelector cameraSelector = CameraSelector.*DEFAULT\_BACK\_CAMERA*;  
  
 ImageAnalysis imageAnalysis = new ImageAnalysis.Builder()  
 .setBackpressureStrategy(ImageAnalysis.*STRATEGY\_KEEP\_ONLY\_LATEST*)  
 .build();  
  
 BarcodeScannerOptions options = new BarcodeScannerOptions.Builder()  
 .setBarcodeFormats(Barcode.*FORMAT\_QR\_CODE*, Barcode.*FORMAT\_AZTEC*, Barcode.*FORMAT\_UPC\_A*)  
 .build();  
 final BarcodeScanner scanner = BarcodeScanning.*getClient*(options);  
  
 imageAnalysis.setAnalyzer(cameraExecutor, imageProxy -> {  
 @androidx.camera.core.ExperimentalGetImage  
 ImageProxy.PlaneProxy[] planes = imageProxy.getPlanes();   
 if(imageProxy.getImage() != null){  
 InputImage image = InputImage.*fromMediaImage*(imageProxy.getImage(), imageProxy.getImageInfo().getRotationDegrees());  
 scanner.process(image)  
 .addOnSuccessListener(barcodes -> {  
 for(Barcode barcode : barcodes){  
 String raw = barcode.getRawValue();  
 if(raw != null){  
 runOnUiThread(() -> tvInfo.setText("QR: " + raw));  
   
 }  
 }  
 })  
 .addOnFailureListener(e -> {})  
 .addOnCompleteListener(task -> imageProxy.close());  
 } else {  
 imageProxy.close();  
 }  
 });  
  
 preview.setSurfaceProvider(previewView.getSurfaceProvider());  
 cameraProvider.bindToLifecycle(this, cameraSelector, preview, imageAnalysis);  
 }  
  
 @Override public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults){  
 if(requestCode==REQ\_CODE){  
 if(grantResults.length>0 && grantResults[0]==PackageManager.*PERMISSION\_GRANTED*) startCamera();  
 else Toast.*makeText*(this,"Permiso de cámara necesario", Toast.*LENGTH\_SHORT*).show();  
 } else super.onRequestPermissionsResult(requestCode, permissions, grantResults);  
 }  
  
 @Override protected void onDestroy(){  
 super.onDestroy();  
 cameraExecutor.shutdown();  
 }  
}

