Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №4 з дисципліни

«Розробка мобільних застосувань під Android»

«Дослідження способів роботи з медіаданими»

Виконав ІП-22 Нижник Дмитро Сергійович _____

(шифр, прізвище, ім'я, по батькові)

Перевірив Орленко Сергій Петрович _____

(прізвище, ім'я, по батькові)

Київ 2025

Завдання лабораторної роботи

БАЗОВЕ (12/20 балів). Написати програму під платформу Андроїд, яка має інтерфейс для запуску аудіо-файлів та відео-файлів. Мінімально інтерфейс має надавати можливість Програвати/Зупиняти/Призупиняти відтворення відео-файлу або аудіо-файлу, який зберігається у внутрішньому сховищі.

ПОВНЕ (20/20). Функціональність базового додатку додатково розширюється наступними можливостями:

- надати вибір типу файлу для відтворення (аудіо або відео) з будь-якого сховища на мобільному пристрої;
- надати вибір завантаження файлу з Інтернету;
- використовувати для реалізації обробки медіа-даних спеціалізовані інструменти (особливу увагу приділити програванню відео).

Хід виконання роботи

У процесі виконання лабораторної роботи було розроблено мобільний додаток на платформі Android, що надає можливість програвати аудіо- та відеофайли як із внутрішнього сховища пристрою, так і завантажені з інтернету. Основою програми наступні класи: MainActivity та PlayerActivity.

Клас MainActivity забезпечує базову взаємодію користувача з додатком. Він містить логіку для вибору джерела медіафайлу: завантаження з інтернету за URL або вибір локального файлу з галереї чи файлового сховища. Завантаження з інтернету реалізовано з використанням сервісу DownloadManager, який дозволяє керувати процесом завантаження файлів.

Клас PlayerActivity відповідає за безпосереднє відтворення медіафайлів. Він підтримує два типи контенту: аудіо та відео, і має відповідні механізми для роботи з кожним із цих типів. Для відтворення відео використовується вбудований компонент VideoView, що забезпечує можливість програвання, паузи, відновлення та перезапуску відеороликів. Для програвання аудіофайлів застосовано клас MediaPlayer. Інтерфейс програвача містить кнопки паузи, відновлення, перезапуску та закриття програвача.

Окрім того, інтерфейс додатку реалізовано у двох режимах: портретному та горизонтальному.

Код програми

MainActivity.kt

```
package com.androidlabs.lab_4

import android.app.AlertDialog
import android.app.DownloadManager
import android.content.Context
import android.content.Intent
import android.net.Uri
import android.os.Bundle
import android.os.Environment
import android.os.Handler
import android.os.Looper
import android.webkit.MimeTypeMap
import android.view.View
import android.widget.EditText
import android.widget.LinearLayout
import android.widget.Toast
import androidx.activity.result.ActivityResultLauncher
import androidx.activity.result.contract.ActivityResultContracts
import androidx.appcompat.app.AppCompatActivity
import java.io.File

class MainActivity : AppCompatActivity() {

    private lateinit var blockUploadFromInternet: LinearLayout
    private lateinit var blockChooseLocal: LinearLayout
    private lateinit var etUrl: EditText
    private lateinit var fileChooserLauncher: ActivityResultLauncher<Intent>
    private lateinit var loadingOverlay: View

    private var downloadId: Long = 0L
    private var currentDownloadName: String = ""

    private val pollHandler = Handler(Looper.getMainLooper())
    private val pollRunnable = object : Runnable {
        override fun run() {
            val dm = getSystemService(Context.DOWNLOAD_SERVICE) as
DownloadManager
            val query = DownloadManager.Query().setFilterById(downloadId)
            val cursor = dm.query(query)

            if (cursor != null && cursor.moveToFirst()) {
                val status =
cursor.getInt(cursor.getColumnIndex(DownloadManager.COLUMN_STATUS))
                when (status) {
                    DownloadManager.STATUS_SUCCESSFUL -> {
                        loadingOverlay.visibility = View.GONE

                        val downloadedUri: Uri? =
dm.getUriForDownloadedFile(downloadId)
                        if (downloadedUri != null) {
                            val resolvedMime =
contentResolver.getType(downloadedUri)
                            val ext = if (!resolvedMime.isNullOrEmpty()) {
                                val extFromMime =
```

```
MimeTypeMap.getSingleton().getExtensionFromMimeType(resolvedMime)
                                if (!extFromMime.isNullOrEmpty()) ".$extFromMime"
else ""
                            } else ""

                            val downloadsDir =
Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS)
                            val originalFile = File(downloadsDir,
currentDownloadName)
                            val newFile = if (ext.isNotEmpty() &&
!currentDownloadName.endsWith(ext, ignoreCase = true)) File(downloadsDir,
currentDownloadName + ext)
                            else originalFile

                            if (originalFile.exists() && originalFile.name !=
newFile.name) {
                                val renamed = originalFile.renameTo(newFile)
                                if (!renamed) {
                                    Toast.makeText(
                                        this@MainActivity,
                                        "Failed to rename downloaded file",
                                        Toast.LENGTH_SHORT
                                    ).show()
                                }
                            }
                            val finalUri = Uri.fromFile(newFile)

                            val isVideo = resolvedMime?.startsWith("video") ==
true

                            val playIntent = Intent(this@MainActivity,
PlayerActivity::class.java).apply {
                                putExtra("media_uri", finalUri)
                                putExtra("isVideo", isVideo)
                                addFlags(Intent.FLAG_GRANT_READ_URI_PERMISSION)
                            }
                            startActivity(playIntent)
                        } else {
                            Toast.makeText(
                                this@MainActivity,
                                "Failed to retrieve downloaded file",
                                Toast.LENGTH_SHORT
                            ).show()
                        }
                        cursor.close()
                        return
                    }
                    DownloadManager.STATUS_FAILED -> {
                        loadingOverlay.visibility = View.GONE
                        Toast.makeText(this@MainActivity, "Download failed",
Toast.LENGTH_SHORT)
                            .show()
                        cursor.close()
                        return
                    }
                }
                cursor.close()
            }
            pollHandler.postDelayed(this, 2000)
```

```
        }

    }

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        supportActionBar?.hide()
        setContentView(R.layout.activity_main)

        etUrl = findViewById(R.id.etUrl)
        blockUploadFromInternet = findViewById(R.id.blockUploadFromInternet)
        blockChooseLocal = findViewById(R.id.blockChooseLocal)
        loadingOverlay = findViewById(R.id.loading_overlay)
        loadingOverlay.visibility = View.GONE

        fileChooserLauncher =

registerForActivityResult(ActivityResultContracts.StartActivityForResult()) {
result ->
                if (result.resultCode == RESULT_OK) {
                    val fileUri: Uri? = result.data?.data
                    if (fileUri != null) {
                        val mimeType = contentResolver.getType(fileUri) ?: ""
                        val isVideo = mimeType.startsWith("video")
                        val intent = Intent(this,
PlayerActivity::class.java).apply {
                            putExtra("media_uri", fileUri)
                            putExtra("isVideo", isVideo)
                        }
                        startActivity(intent)
                    }
                }
            }

        blockUploadFromInternet.setOnClickListener {
            val url = etUrl.text.toString().trim()
            if (url.isEmpty()) {
                showAlert("Error", "URL field is empty!")
                return@setOnClickListener
            }
            if (!url.startsWith("http")) {
                showAlert("Error", "Invalid URL!")
                return@setOnClickListener
            }
            try {
                val dm = getSystemService(Context.DOWNLOAD_SERVICE) as
DownloadManager
                val downloadUri: Uri = Uri.parse(url)
                currentDownloadName = System.currentTimeMillis().toString()
                val request = DownloadManager.Request(downloadUri)
                    .setAllowedNetworkTypes(
                        DownloadManager.Request.NETWORK_WIFI or
                                DownloadManager.Request.NETWORK_MOBILE
                    )
                    .setMimeType("")
                    .setTitle(currentDownloadName)
                    .setDescription("Downloading media file")
                    .setNotificationVisibility(
```

```
DownloadManager.Request.VISIBILITY_VISIBLE_NOTIFY_COMPLETED
                    )
                    .setAllowedOverRoaming(false)

.setDestinationInExternalPublicDir(Environment.DIRECTORY_DOWNLOADS,
currentDownloadName)
                downloadId = dm.enqueue(request)
                loadingOverlay.visibility = View.VISIBLE
                Toast.makeText(this, "Download started...",
Toast.LENGTH_SHORT).show()
                pollHandler.postDelayed(pollRunnable, 2000)
            } catch (e: Exception) {
                showAlert("Error", "Failed to start download:
${e.localizedMessage}")
            }
        }

        blockChooseLocal.setOnClickListener {
            val options = arrayOf("Gallery", "File Storage")
            AlertDialog.Builder(this)
                .setTitle("Select Source")
                .setItems(options) { dialog, which ->
                    when (which) {
                        0 -> {
                            val intent = Intent(Intent.ACTION_PICK).apply { type
= "video/*" }
                            fileChooserLauncher.launch(intent)
                        }
                        1 -> {
                            val intent =
Intent(Intent.ACTION_OPEN_DOCUMENT).apply {
                                addCategory(Intent.CATEGORY_OPENABLE)
                                type = "*/*"
                                val mimeTypes = arrayOf("audio/*", "video/*")
                                putExtra(Intent.EXTRA_MIME_TYPES, mimeTypes)
                            }
                            fileChooserLauncher.launch(intent)
                        }
                    }
                }
                .setNegativeButton("Cancel") { dialog, _ -> dialog.dismiss() }
                .show()
        }

    }

    override fun onDestroy() {
        super.onDestroy()
        pollHandler.removeCallbacks(pollRunnable)
    }

    private fun showAlert(title: String, message: String) {
        AlertDialog.Builder(this)
            .setTitle(title)
            .setMessage(message)
            .setPositiveButton("OK") { dialog, _ -> dialog.dismiss() }
            .show()
    }
```

```
}
```

PlayActivity.kt

```
package com.androidlabs.lab_4

import android.app.AlertDialog
import android.graphics.BitmapFactory
import android.media.MediaMetadataRetriever
import android.media.MediaPlayer
import android.net.Uri
import android.os.Bundle
import android.os.Handler
import android.os.Looper
import android.widget.Button
import android.widget.ImageView
import android.widget.SeekBar
import android.widget.TextView
import android.widget.VideoView
import androidx.appcompat.app.AppCompatActivity

class PlayerActivity : AppCompatActivity() {

    private var mediaPlayer: MediaPlayer? = null
    private var videoView: VideoView? = null
    private lateinit var seekBar: SeekBar
    private lateinit var btnPause: Button
    private lateinit var btnResume: Button
    private lateinit var btnRestart: Button
    private lateinit var btnExit: Button
    private lateinit var coverImage: ImageView
    private lateinit var tvTime: TextView

    private var isUserSeeking = false
    private val handler = Handler(Looper.getMainLooper())
    private val updateSeekBarRunnable = object : Runnable {
        override fun run() {
            if (!isUserSeeking) {
                if (isVideo) {
                    videoView?.let { vv ->
                        val pos = vv.currentPosition
                        seekBar.progress = pos
                        tvTime.text = formatTime(pos)
                    }
                } else {
                    mediaPlayer?.let { mp ->
                        val pos = mp.currentPosition
                        seekBar.progress = pos
                        tvTime.text = formatTime(pos)
                    }
                }
            }
            handler.postDelayed(this, 500)
        }
    }
```

```
    private var mediaUri: Uri? = null
    private var isVideo: Boolean = false

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        supportActionBar?.hide()
        setContentView(R.layout.activity_player)

        videoView = findViewById(R.id.videoView)
        seekBar = findViewById(R.id.seekBar)
        btnPause = findViewById(R.id.btnPause)
        btnResume = findViewById(R.id.btnResume)
        btnRestart = findViewById(R.id.btnRestart)
        btnExit = findViewById(R.id.btnExit)
        coverImage = findViewById(R.id.coverImage)
        tvTime = findViewById(R.id.tvTime)

        mediaUri = intent.getParcelableExtra("media_uri")
        isVideo = intent.getBooleanExtra("isVideo", false)

        if (mediaUri == null) {
            showAlert("Error", "Media file not provided!")
            return
        }

        if (isVideo)
            setupVideo(mediaUri!!)
        else
            setupAudio(mediaUri!!)

        btnPause.setOnClickListener { pauseMedia() }
        btnResume.setOnClickListener { resumeMedia() }
        btnRestart.setOnClickListener { restartMedia() }
        btnExit.setOnClickListener { finish() }

        seekBar.setOnSeekBarChangeListener(object :
SeekBar.OnSeekBarChangeListener {
            override fun onProgressChanged(sb: SeekBar?, progress: Int, fromUser:
Boolean) {
                if (fromUser) {
                    tvTime.text = formatTime(progress)
                }
            }
            override fun onStartTrackingTouch(sb: SeekBar?) {
                isUserSeeking = true
                handler.removeCallbacks(updateSeekBarRunnable)
            }
            override fun onStopTrackingTouch(sb: SeekBar?) {
                isUserSeeking = false
                val pos = sb?.progress ?: 0
                if (isVideo)
                    videoView?.seekTo(pos)
                else
                    mediaPlayer?.seekTo(pos)
                handler.post(updateSeekBarRunnable)
            }
        })
    }
```

```
    private fun setupVideo(uri: Uri) {
        videoView?.visibility = android.view.View.VISIBLE
        coverImage.visibility = android.view.View.GONE
        videoView?.apply {
            setVideoURI(uri)
            setOnPreparedListener { mp ->
                seekBar.max = mp.duration
                start()
                updateButtonStates(true)
                handler.post(updateSeekBarRunnable)
            }
            setOnCompletionListener { updateButtonStates(false) }
        }
    }

    private fun setupAudio(uri: Uri) {
        videoView?.visibility = android.view.View.GONE
        coverImage.visibility = android.view.View.VISIBLE
        val retriever = MediaMetadataRetriever()
        try {
            retriever.setDataSource(this, uri)
            val art = retriever.embeddedPicture
            if (art != null) {
                val bitmap = BitmapFactory.decodeByteArray(art, 0, art.size)
                coverImage.setImageBitmap(bitmap)
            } else {

coverImage.setBackgroundColor(resources.getColor(android.R.color.darker_gray))
            }
        } catch (e: Exception) {

coverImage.setBackgroundColor(resources.getColor(android.R.color.darker_gray))
        } finally {
            retriever.release()
        }
        mediaPlayer = MediaPlayer().apply {
            setDataSource(this@PlayerActivity, uri)
            prepareAsync()
            setOnPreparedListener { mp ->
                seekBar.max = mp.duration
                start()
                updateButtonStates(true)
                handler.post(updateSeekBarRunnable)
            }
            setOnCompletionListener { updateButtonStates(false) }
        }
    }

    private fun pauseMedia() {
        if (isVideo)
            videoView?.pause()
        else
            mediaPlayer?.pause()
        updateButtonStates(false)
    }

    private fun resumeMedia() {
        if (isVideo)
            videoView?.start()
```

```
        else
            mediaPlayer?.start()
        updateButtonStates(true)
    }

    private fun restartMedia() {
        if (isVideo) {
            videoView?.seekTo(0)
            videoView?.start()
            updateButtonStates(true)
        } else {
            mediaPlayer?.seekTo(0)
            mediaPlayer?.start()
            updateButtonStates(true)
        }
    }

    private fun updateButtonStates(isPlaying: Boolean) {
        if (isPlaying) {
            btnPause.setBackgroundResource(R.drawable.pause)
            btnResume.setBackgroundResource(R.drawable.play_inactive)
        } else {
            btnPause.setBackgroundResource(R.drawable.pause_inactive)
            btnResume.setBackgroundResource(R.drawable.play)
        }
    }

    override fun onDestroy() {
        super.onDestroy()
        handler.removeCallbacks(updateSeekBarRunnable)
        mediaPlayer?.release()
        mediaPlayer = null
    }

    private fun formatTime(milliseconds: Int): String {
        val totalSeconds = milliseconds / 1000
        val seconds = totalSeconds % 60
        val minutes = (totalSeconds / 60) % 60
        val hours = totalSeconds / 3600
        return if (hours > 0)
            String.format("%d:%02d:%02d", hours, minutes, seconds)
        else
            String.format("%02d:%02d", minutes, seconds)
    }

    private fun showAlert(title: String, message: String) {
        AlertDialog.Builder(this)
            .setTitle(title)
            .setMessage(message)
            .setPositiveButton("OK") { dialog, _ ->
                dialog.dismiss()
                finish()
            }
            .show()
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#CCCCCC"
    android:fitsSystemWindows="true">

    <LinearLayout
        android:id="@+id/page_title"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="15dp"
        android:paddingStart="15dp"
        android:paddingEnd="8dp"
        android:paddingBottom="20dp"
        android:orientation="horizontal"
        android:gravity="center_vertical"
        android:background="@color/black">

        <TextView
            android:id="@+id/page_title_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="30sp"
            android:textStyle="bold"
            android:text="@string/app_header"
            android:textColor="@color/white"/>

    </LinearLayout>

    <LinearLayout
        android:id="@+id/main_content"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@id/page_title"
        android:orientation="vertical"
        android:padding="15dp"
        android:background="#707070">

        <EditText
            android:id="@+id/etUrl"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="@string/paste_URL"
            android:textColorHint="#CCCCCC"
            android:textStyle="italic"
            android:inputType="textUri"
            android:textSize="18sp"
            android:background="@drawable/edittext_underline"
            android:padding="8dp" />

        <LinearLayout
            android:id="@+id/blockUploadFromInternet"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_marginTop="15dp"
```

```
            android:layout_marginStart="2dp"
            android:layout_marginEnd="2dp"
            android:clickable="true"
            android:focusable="true"
            android:orientation="vertical"
            android:gravity="center"
            android:layout_weight="1"
            android:background="@drawable/dashed_border">

            <ImageView
                android:id="@+id/iconUploadInternet"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:src="@drawable/baseline_cloud_upload_24" />

            <TextView
                android:id="@+id/textUploadInternet"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/upload_file"
                android:textColor="@android:color/white"
                android:textSize="16sp"
                android:layout_marginTop="8dp" />

        </LinearLayout>

        <TextView
            android:id="@+id/or_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/or"
            android:layout_gravity="center"
            android:textColor="@android:color/white"
            android:textSize="22sp"
            android:layout_marginTop="8dp" />

        <LinearLayout
            android:id="@+id/blockChooseLocal"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_marginTop="8dp"
            android:layout_marginBottom="15dp"
            android:layout_marginStart="2dp"
            android:layout_marginEnd="2dp"
            android:clickable="true"
            android:focusable="true"
            android:orientation="vertical"
            android:gravity="center"
            android:layout_weight="3"
            android:background="@drawable/dashed_border">

            <ImageView
                android:id="@+id/iconChooseLocal"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:src="@drawable/baseline_drive_folder_upload_24"/>

            <TextView
                android:id="@+id/textChooseLocal"
```

```
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/select_file"
                android:textColor="@android:color/white"
                android:textSize="16sp"
                android:layout_marginTop="8dp" />

        </LinearLayout>

    </LinearLayout>

    <FrameLayout
        android:id="@+id/loading_overlay"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#88000000"
        android:visibility="gone"
        android:clickable="true"
        android:focusable="true">

        <ProgressBar
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"/>

    </FrameLayout>

</RelativeLayout>
```

activity_player.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/playerLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    android:background="#707070">

    <LinearLayout
        android:id="@+id/page_title"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="18dp"
        android:paddingStart="15dp"
        android:paddingEnd="8dp"
        android:paddingBottom="20dp"
        android:orientation="horizontal"
        android:gravity="center_vertical"
        android:background="@color/black"
        app:layout_constraintTop_toTopOf="parent">

        <TextView
            android:id="@+id/page_title_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="30sp"
            android:layout_weight="1.5"
```

```
            android:textStyle="bold"
            android:text="@string/app_header"
            android:textColor="@color/white"/>

        <Button
            android:id="@+id/btnExit"
            android:layout_width="45dp"
            android:layout_height="45dp"
            android:background="@drawable/back"
            android:layout_marginEnd="8dp"/>

    </LinearLayout>

    <FrameLayout
        android:id="@+id/mediaContainer"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:layout_constraintTop_toBottomOf="@id/page_title"
        app:layout_constraintBottom_toTopOf="@id/tvTime"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_margin="16dp">

        <VideoView
            android:id="@+id/videoView"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_gravity="center"
            android:visibility="gone"/>

        <ImageView
            android:id="@+id/coverImage"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:scaleType="centerCrop"
            android:background="#CCCCCC"
            android:visibility="gone"/>

    </FrameLayout>

    <TextView
        android:id="@+id/tvTime"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="00:00"
        android:textSize="16sp"
        app:layout_constraintTop_toBottomOf="@id/mediaContainer"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_marginTop="8dp" />

    <SeekBar
        android:id="@+id/seekBar"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:progressTint="@android:color/white"
        android:thumbTint="@android:color/white"
        app:layout_constraintTop_toBottomOf="@id/tvTime"
        app:layout_constraintBottom_toTopOf="@id/buttonContainer"
```

```
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_marginHorizontal="16dp"
        android:layout_marginBottom="8dp"/>

    <LinearLayout
        android:id="@+id/buttonContainer"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_marginBottom="16dp">

        <Button
            android:id="@+id/btnPause"
            android:layout_width="35dp"
            android:layout_height="35dp"
            android:background="@drawable/pause"
            android:layout_marginEnd="20dp"/>

        <Button
            android:id="@+id/btnResume"
            android:layout_width="30dp"
            android:layout_height="30dp"
            android:background="@drawable/play"
            android:layout_marginEnd="10dp"/>

        <Button
            android:id="@+id/btnRestart"
            android:layout_width="42dp"
            android:layout_height="42dp"
            android:background="@drawable/restart"/>

    </LinearLayout>

</androidx.constraintlayout.widget.ConstraintLayout>
```