Assignment - VI

Problem statement :-

Design a mobile app for media player to store data using internal or external storage.

Pre requisite :-

1) Basic concept of internal or external storage.
2) Basic concept of internal & external memory.

S/W & H/W :-

Android Studio, 4 GB RAM, 64 bit OS.

objective :-

Implement app to store data using internal or external storage.

Outcome :-

After completion of this assignment student are able to implement app to store data using internal or external storage.

Theory :-

Actually, the first thing you do is create an activity. These are where all the action happens, because they are the screen that allow user to interact with your app. In short activities are one of the basic building blocks of Android application. The process for creating, starting & stopping an activity & handle navigation bet^{n} activities. The various stages in lifecycle of an activity & how to handle each stage gracefully.
The way to manage configurations changes & persist data within your activity.

- Android Preference Example :-

Android shared preference is used to store & retrive primitive information. In Android, String, long, integer, number, etc. are considered as primitive data types.

It is used to store data in key & value pair so that we can retrive the value on basis of key.

Android provides many kinds of storage for applications to store their data. These storage places are shared preference, internal & external storage, SQLite storage, and storage via network connection. It is widely used to get information from user such as in settings.

- Android Internal Storage Example:

We are able to save or read data from device internal memory. FileInputStream & FileOutputStream classes are used to read & write data into file.

In order to use internal storage to write some data in file call the openFileOutput() method with the name of file & mode. The mode could be private, public, etc. The syntax is,

```
FileOutputStream fout = openFileOutput(
"file name here", MODE_WORLD-READABLE);
```

Apart from the methods of read & close; there are other method provided by FileInputStream class for better reading files. These methods are listed below.

1) available()

This method returns an estimated number of bytes that can be read or skipped without blocking for more input.

2) getchannel():

This method returns a read only file channel that shares position within this stream.

3) getFD():

This method returns the underlying file descriptor.

4) read(byte[] buffer, int byte offset, int byte count)

This method reads at most length bytes from this stream & stores them in the byte array b starting at offset.

5 Android External Storage:

Like internal storage, we are able to save or read data from the device's external memory such as sdcard. The FileInputStream & FileOutputStream classes are used to read & write data into the file.

conclusion :-

We successfully implement the media player where we can play the songs from internal or external memory.

MainActivity.java

```
package com.example.music;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.os.Environment;
import android.provider.ContactsContract;
import android.provider.MediaStore;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;

import com.karumi.dexter.Dexter;
import com.karumi.dexter.PermissionToken;
import com.karumi.dexter.listener.PermissionDeniedResponse;
import com.karumi.dexter.listener.PermissionGrantedResponse;
import com.karumi.dexter.listener.PermissionRequest;
import com.karumi.dexter.listener.single.PermissionListener;

import java.io.File;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {
private ListView listView;
Toolbar toolbar;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        listView=findViewById(R.id.list_songs);
        toolbar=findViewById(R.id.toolbar1);
        setSupportActionBar(toolbar);
        checkpermission();
    }
    public void checkpermission()
    {

Dexter.withActivity(this).withPermission(Manifest.permission.READ_EXTERNAL_STORAGE).with
Listener(new PermissionListener() {
          @Override
          public void onPermissionGranted(PermissionGrantedResponse response) {
              display();
          }
```

```
            @Override
            public void onPermissionDenied(PermissionDeniedResponse response) {

            }

            @Override
            public void onPermissionRationaleShouldBeShown(PermissionRequest permission,
PermissionToken token) {
             token.continuePermissionRequest();
            }
        }).check();
    }

    private void display() {
        final ArrayList<File>songs=findsong(Environment.getExternalStorageDirectory());
        String[]items=new String[songs.size()];
        for (int i = 0; i <songs.size() ; i++) {
           items[i]=songs.get(i).getName();
        }
        ArrayAdapter <String>adapter=new
ArrayAdapter<String>(this,android.R.layout.simple_list_item_1,items);
        listView.setAdapter(adapter);
        listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> adapterView, View view, int i, long
l) {
                String name=listView.getItemAtPosition(i).toString();
                startActivity(new
Intent(MainActivity.this,player.class).putExtra("songs",songs)
                .putExtra("songname",name).putExtra("pos",i));
            }
        });

    }

    private ArrayList<File> findsong(File f) {
        ArrayList<File>arrayList=new ArrayList<>();
        File[]files=f.listFiles();
        for (File single:files
             ) {
            if (single.isDirectory())
            {
                arrayList.addAll(findsong(single));
            }
            else {
                if(single.getName().endsWith(".mp3") ||
single.getName().endsWith(".m4a") || single.getName().endsWith(".wav") ||
single.getName().endsWith(".m4b"))
                {
                    arrayList.add(single);
                }
            }

        }
        return arrayList;
    }

    }
```

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"
    tools:context=".MainActivity">
    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar1"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:background="@color/colorAccent"
        android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
        app:layout_constraintBottom_toTopOf="@+id/image"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0"
        app:popupTheme="@style/ThemeOverlay.AppCompat.Dark"
        app:title="MY Player" />
<ListView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/list_songs"
    />
</LinearLayout>
```

Player.java

```
package com.example.music;

import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.AppCompatSeekBar;
import androidx.appcompat.widget.Toolbar;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.graphics.PorterDuff;
import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.view.View;
import android.widget.ImageButton;
import android.widget.SeekBar;
import android.widget.TextView;

import java.io.File;
import java.util.ArrayList;

import static android.os.SystemClock.sleep;

public class player extends AppCompatActivity implements View.OnClickListener {
private ImageButton play,prev,next;
```

```
TextView curtime,maxtime,name;
private int pos;
Thread t;
Toolbar toolbar;
private AppCompatSeekBar seekbar;
    ArrayList<File>mysongs;
MediaPlayer mediaplayer;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_player);
        play=findViewById(R.id.play);
        prev=findViewById(R.id.prev);
        next=findViewById(R.id.next);
        seekbar=findViewById(R.id.seek);
        curtime=findViewById(R.id.curtime);
        maxtime=findViewById(R.id.maxtime);
        name=findViewById(R.id.name);
        toolbar=findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        getSupportActionBar().setDisplayShowHomeEnabled(true);
        if (mediaplayer != null) {
            mediaplayer.stop();
        }
        t=new Thread(new Runnable() {
            @Override
            public void run() {
                while(mediaplayer!=null)
                {

                    if (mediaplayer.isPlaying())
                    {

                        Message msg = new Message();
                        msg.what = mediaplayer.getCurrentPosition();
                        handler.sendMessage(msg);
                        sleep(1000);
                    }
                }
            }
        });

        Intent i=getIntent();
        Bundle bundle=i.getExtras();
        mysongs=(ArrayList)i.getParcelableArrayListExtra("songs");
        String sname=i.getStringExtra("name");
         pos=bundle.getInt("pos");
        Uri u= Uri.parse(mysongs.get(pos).toString());
        String same = mysongs.get(pos).getName().replace(".mp3", "").replace(".m4a",
"").replace(".wav", "").replace(".m4b", "");
        name.setText(same);
        mediaplayer=MediaPlayer.create(getApplicationContext(),u);

        mediaplayer.start();
        play.setOnClickListener(this);
        prev.setOnClickListener(this);
        next.setOnClickListener(this);
       seekbar.setMax(mediaplayer.getDuration());
       String max=createTimeLabel(mediaplayer.getDuration());
       maxtime.setText(max);
       t.start();

seekbar.getProgressDrawable().setColorFilter(getResources().getColor(R.color.colorPrimar
y), PorterDuff.Mode.MULTIPLY);
        seekbar.getThumb().setColorFilter(getResources().getColor(R.color.colorPrimary),
PorterDuff.Mode.SRC_IN);
        seekbar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
```

```
        @Override
        public void onProgressChanged(SeekBar seekBar, int i, boolean b) {

        }

        @Override
        public void onStartTrackingTouch(SeekBar seekBar) {

        }

        @Override
        public void onStopTrackingTouch(SeekBar seekBar) {
        mediaplayer.seekTo(seekBar.getProgress());
        }
    });
}

@Override
public void onClick(View view) {
   switch(view.getId())
   {
       case R.id.play :
           pause();
           break;
       case R.id.prev :
           previoussong();
           break;
       case R.id.next :
           nextsong();
   }
}

private void pause() {
    if (mediaplayer.isPlaying()){
        mediaplayer.pause();
        play.setImageResource(R.drawable.play);
    }
    else
    {
        mediaplayer.start();
        play.setImageResource(R.drawable.pause);
    }
}
private void previoussong()
{
    if (pos <= 0) {
        pos = mysongs.size() - 1;
    } else {
        pos--;
    }

    initPlayer(pos);
}
private void nextsong(){
    play.setImageResource(R.drawable.pause);
    if (pos < mysongs.size() - 1) {
        pos++;
    } else {
        pos = 0;

    }
    initPlayer(pos);

}
public String createTimeLabel(int duration) {
    String timeLabel = "";
```

```
        int min = duration / 1000 / 60;
        int sec = duration / 1000 % 60;

        timeLabel += min + ":";
        if (sec < 10) timeLabel += "0";
        timeLabel += sec;

        return timeLabel;
    }
    private void initPlayer(final int position) {

        if (mediaplayer != null && mediaplayer.isPlaying()) {
            mediaplayer.reset();
        }

        String sname = mysongs.get(position).getName().replace(".mp3",
"").replace(".m4a", "").replace(".wav", "").replace(".m4b", "");
        name.setText(sname);
        Uri songResourceUri = Uri.parse(mysongs.get(position).toString());

        mediaplayer = MediaPlayer.create(getApplicationContext(), songResourceUri); //
        mediaplayer.setOnPreparedListener(new MediaPlayer.OnPreparedListener() {
            @Override
            public void onPrepared(MediaPlayer mp) {
                String totalTime = createTimeLabel(mediaplayer.getDuration());
                maxtime.setText(totalTime);
                seekbar.setMax(mediaplayer.getDuration());
                mediaplayer.start();
                play.setImageResource(R.drawable.pause);

            }
        });
        mediaplayer.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {
            @Override
            public void onCompletion(MediaPlayer mp) {
                int curSongPoition = position;

                if (curSongPoition < mysongs.size() - 1) {
                    curSongPoition++;
                    initPlayer(curSongPoition);
                } else {
                    curSongPoition = 0;
                    initPlayer(curSongPoition);
                }


            }
        });
    }
    @SuppressLint("HandlerLeak")
    private Handler handler = new Handler() {
        @Override
        public void handleMessage(Message msg) {
//            Log.i("handler ", "handler called");
            int current_position = msg.what;
            seekbar.setProgress(current_position);
            String cTime = createTimeLabel(current_position);
            curtime.setText(cTime);
        }
    };
    @Override
    public boolean onSupportNavigateUp() {
        onBackPressed();
        return true;
    }
```

```
    @Override
    public void onBackPressed() {
        mediaplayer.stop();
        super.onBackPressed();
    }
}
```

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"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"

    tools:context=".player">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:background="@color/colorAccent"
        android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
        app:layout_constraintBottom_toTopOf="@+id/image"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0"
        app:popupTheme="@style/ThemeOverlay.AppCompat.Dark"
        app:title="MY Player" />

    <ImageView
        android:id="@+id/image"
        android:layout_width="300dp"
        android:layout_height="300dp"
        android:fitsSystemWindows="true"
        android:scaleType="centerCrop"
        android:src="@drawable/music"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.229" />

    <androidx.appcompat.widget.AppCompatSeekBar
        android:id="@+id/seek"
        android:layout_width="match_parent"
        android:layout_height="20dp"
        android:layout_marginLeft="32dp"
        android:layout_marginRight="32dp"
        app:layout_constraintBottom_toTopOf="@id/relative"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toBottomOf="@id/image"
        app:layout_constraintVertical_bias="0.797" />
```

```
<RelativeLayout
    android:id="@+id/relative"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/image"
    app:layout_constraintVertical_bias="0.756">

    <ImageButton
        android:id="@+id/prev"
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:layout_marginRight="30dp"
        android:layout_toLeftOf="@id/play"
        android:src="@drawable/prev" />

    <ImageButton
        android:id="@+id/play"
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:layout_centerHorizontal="true"
        android:src="@drawable/pause" />

    <ImageButton
        android:id="@+id/next"
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:layout_marginLeft="30dp"
        android:layout_toRightOf="@id/play"
        android:src="@drawable/next" />


</RelativeLayout>

<TextView
    android:id="@+id/curtime"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="00:00"
  android:layout_marginLeft="5dp"
    app:layout_constraintRight_toLeftOf="@id/seek"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintBottom_toBottomOf="@+id/seek"
    app:layout_constraintTop_toTopOf="@+id/seek"
    tools:layout_editor_absoluteX="-3dp" />

<TextView
    android:id="@+id/maxtime"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="00:00"
    android:layout_marginRight="5dp"
    app:layout_constraintBottom_toBottomOf="@+id/seek"
    app:layout_constraintTop_toTopOf="@+id/seek"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintLeft_toRightOf="@id/seek"
    tools:layout_editor_absoluteX="379dp" />

<TextView
    android:id="@+id/name"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
```

```
        android:text="SongName"
        app:layout_constraintBottom_toTopOf="@+id/seek"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/image" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

MY Player

Vitthala Konta Zenda-(Mr-Jatt.com).mp3

Wicked World.mp3

Forgotten Souls.mp3

Im a Bad Man.mp3

Devils Gonna Come.mp3

Jet Black Hearse.mp3

Im Hunted.mp3

Don t Let Me Go - Cigarettes After Sex.mp3

Roar - Katy Perry 🎵.mp3

Silicon Valley Season One Score Suite.mp3

La casa de papel Soundtrack Cecilia Krull - My life is going on.mp3

8815_download_best_church_bell_ringtone.mp3

Fatteshikast Theme.mp3

Silicon Valley Season One Score Suite (1).mp3

Silicon Valley Show Finale Score (Guess We'll Find Out - Jeff Cardoni).mp3

Johnny Cash - Ain't No Grave.mp3

MY Player

11 bappa morya re

0:04 3:21

MY Player

11 bappa morya re

1:16 3:21