A

PROJECT REPORT

ON

MUSIC PLAYER APP

Submitted by

Yash Chutake (1921321242087)
Rushikesh Dandge (1921321242136)
Pankaj Bedre (1921321242089)

Under Guidance of

Prof. Priyanka Dekshmukh.

Department of Computer Science and Engineering
Jawaharlal Nehru Engineering College, Aurangabad
(Affiliated to Dr. Babasaheb Ambedkar Technological University, Lonere)
(Year 2021-22)

Jawaharlal Nehru Engineering College, Aurangabad

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that Rushikesh Dandge, Yash Chutake, Pankaj Bedre of TY II (Computer Science and Engineering), Roll No. 1921321242136, 1921321242087, 1921321242089, has successfully completed project on **"Music Player Application"** under the guidance of Prof. Priyanka Deshmukh submitted the same during the academic year 2021-2022 towards the partial fulfillment of degree of B. Tech. (Computer Science and Engineering) from Jawaharlal Nehru Engineering College, Aurangabad (An institute affiliated to Dr. Babasaheb Ambedkar Technological University, Lonere, MH, India).

Prof. Priyanka Deshmukh — Project Guide

Dr. V Musande. — HOD CSE

Date: 25 June 2022

Acknowledgement

As we write this acknowledgement, we must clarify that this is not just a formal acknowledgement but also a sincere note of thanks and regard from my side. We feel a deep sense of gratitude and affection for those who were associated with this seminar. Without their co-operation and guidance this seminar could not have been conducted properly.

We also indebted to my friends and family for their constant support and their priceless reviews which helped me to take this mini project to its current level.

Yash Chutake
Rushikesh Dandge
Pankaj Bedre

CONTENTS

Sr. no.	Index	Page no.
1	INTRODUCTION	5
	i. Build developing environment of android	
	ii. The design principle of android application	
	iii. Function and sructure design of android system	
2	REQUREMENT ANALYSIS OF SYSTEM	8
	i. The feasibility analysis	
	ii. Economic feasibility	
	iii. Technical feasibility	
	iv. Social feasibility	
	v. Saturation overview	
3	SYSTEM DESIGN	11
	i. Introduction of player project	
	ii. Part of function design	
	iii. Data storage	
4	FLOWCHART	13
5	APPLICATION UI	14
6	SOURCE CODE	15
7	CONCLUSION	24
	REFERENCES	25

INTRODUCTION

Android is open source code mobile phone operating system that comes out by Google. Music player in this project is application software based on Google Android. Music is one of the best ways to relieve pressure in stressful modern society life. The purpose of this project is to develop a player which can play the mainstream file format. To browse and query the storage space as well as operation of playing can be realised. Meanwhile, this software can play, pause and select songs with latest button and next button according to sets requirement as well as set up songs.

BUILD DEVELOPING ENEVIRONMENT OF ANDROID

The application of android need to run based on Android environment. The following is the configuration requirement and installation steps of Android development environment:

The required software of the developing environment

- Operation system: Windows 10,Linux
- Software：Android SDK(Software Development Kit)、ADT(Android Development Tool)
- JDK：Java Runtime Environment virtual machine、Java Development Kit(JDK)

Installation steps of the developing environment

- Step 1: install the Java virtual machine JDK version - 7
- Step 2: install the Android SDK: first download the Android SDK
- Download address: http://developer-android-com/sdk/index-html
- Input SDK tools path in the SDK location: D: \ android \ software \ android

SDK– Windows and click OK.

 The Android environment is set up successfully.

Hardware Requirements to Run the APP

 This APP meets minimum hardware Requirements:

 >1 GHz Processor

 >512MB of RAM

 >50MB of Internal Storage

THE DESIGN PRINCIPLE OF ANDROID APPLICATION

Twice the result with half the effort will get if an overall study of the principles done before the design and follow them in the operation. The principle of software design mainly includes the following points:

(1) Reliability

The reliability of the software design must be determined. The reliability of the software system refers to the ability to avoid fault occurred in the process of system running, as well as the ability to remedy troubles once the fault occurs.

(2) Reusability

Look for commonness of similar codes, and come out new method abstractly and reasonably. Pay attention to the generic design.

(3) Understandability

The understandability of software not only require clear and readable document, but the simplified structure of software itself, which requires the designer possess keen insight and creativity, and know well about the design objects.

(4) Simple program

To keep the program simple and clear, good programmers can use simple program to solve complex problems.

(5) Testability

Testability means that the created system has a proper data collection to conduct a comprehensive test of the entire system.

(6) The Open-Closed Principal

Module is extensible but cannot be modified. That is to say, extension is open to the existing code in order to adapt to the new requirements. While modify is closed to the categories. Once the design is completed, the categories cannot be modified.

FUNCTION AND STRUCTURE DESIGN OF ANDROID SYSTEM

This system adopts the modularized program design, and system function is correspondingly divided into function modules, the main modules include:

(1)UI function module design of mobile terminal:

The index screen, play screen, music adding page, file management page are realized.

(2) Backstage function module design of mobile terminal:

The specific function, music file data storage function and other function are implemented.

REQUIREMENT ANALYSIS OF SYSTEM

The feasibility analysis:

This section verified that it is feasible to add music player on the Android system from the aspects of economic, technical and social feasibility.

Economic feasibility:

To design Android mobile phone music player as long as a computer has the Android development and the application development of Android is free. In addition, mobile phone music player is basic needs for public. The information that which functions are necessary form all the consumers , which functions are needed for some people, and which features are seldom to use is easy to understand. And a lot of research is eliminated, thus saved the spending. Therefore, the whole process of development doesn't need to spend any money that is economic feasibility.

.

Technical feasibility:

To design a music player which meets the basic requirements, a deep understand of JAVA language, the Android system architecture, application of framework and other technical knowledge are needed.(framework is the core of the application, and rules that all the programmers participating in the development must abide by). Based on the related technology information and resources for Android on the market, and equipped with technical personnel of technology and the spirit of willing to learn, the technology is feasible.

Social Feasibility

With the rapid development of the mobile phone market, all kinds of audio and video resources are widely circulated on the Internet. These resources seem

ordinary, but have gradually become an indispensable part of people life, which derived the development of all kinds of mobile phone player. But a lot of players devoted to fancy appearance, strong function causing a lot of wasted resources to the user's mobile phone and bringing a lot of inconvenience to the user as multitasking operation is needed. Some functions are useless to ordinary people. Powerful player is a good thing, but a lot of functions are actually useless for most users. Aimed at these problems, developing multiplied audio player which owns the features of simplified functions, common play function, meeting the needs of most users, less required memory and high quality of playing music, maximizes the optimization in performance.

Saturation Overview:

This section describes requirements of the system based on basic control functions of players, and system setup function of the player according to research results of the project demand.

According to the research results of project demand, the basic requirements of project system and its function structure are presented. And describe the demand of the system through the different angles. The project is divided into the following parts by using diagram: the basic control functions of the player, the playlist management function of the player and system setting function of the player. The player interface requires rational layout, comfortable color, friendly control buttons and concise and beautiful images. According to the Android system requires, the non- response time is 5 seconds.

The following is requirements in the music player development application:

Application response time shall not exceed 5 seconds in music playing.

Application response time shall not exceed 5 seconds as the music is suspended.

Application response time shall not exceed 5 seconds as the music is stopped.

Application response time shall not exceed 5 seconds as Move Next/Move Previous music.

Application response time shall not exceed 5 seconds as system listing is required.

SYSTEM DESIGN

The App Starting module of the player in the project is introduced, as well as the Android engineering program structure, etc.

Any AppStarting needs AndroidManifest. XML file to start. And any new project content will automatically generate an AndroidManifest. XML file. Configuration files are the core of the whole program, which contains the Android SDK version, and the default Activity in program running. The systems will automatically looking for a logo in AndroidManifest to react the corresponding operation when any component of the program triggers events.

To define the system, the first thing is launching the Activity: Android Activity.

There are properties such as action and category in < intent - filter >. Most of these are the default values of the system. Setting the action and category realize the switch between different Activities. When any components of the program is about to use, declaration must be in the Android Manifest. Xml files. To be clear that authorities must be illustrated as the statement of provider. Each component has a lot of attributes; the program will define different attributes according to different needs.

The basic structure content of Android project includes: the SRC (source code), gen (constant that Android system automatically generates), res (resource file), and the layout of file and pictures in the main storage program interface.

Part of the function design:

The main play interface design.

Convenience and practical should be fully considered in the design of the main interface. Every Android interface is a visual interface, which has its unique layout configuration files. We can configure various layout and resources files according to the requirements, such as images, text and color reference, which can form different visual interface and glaring effect.

Interface design of adding songs.

There are no corresponding songs for the first time login entering the program; users need to add songs to play. Therefore, you need to enter the adding songs' interface. The empty playlist needs to add songs which can choose from the SD card to add.

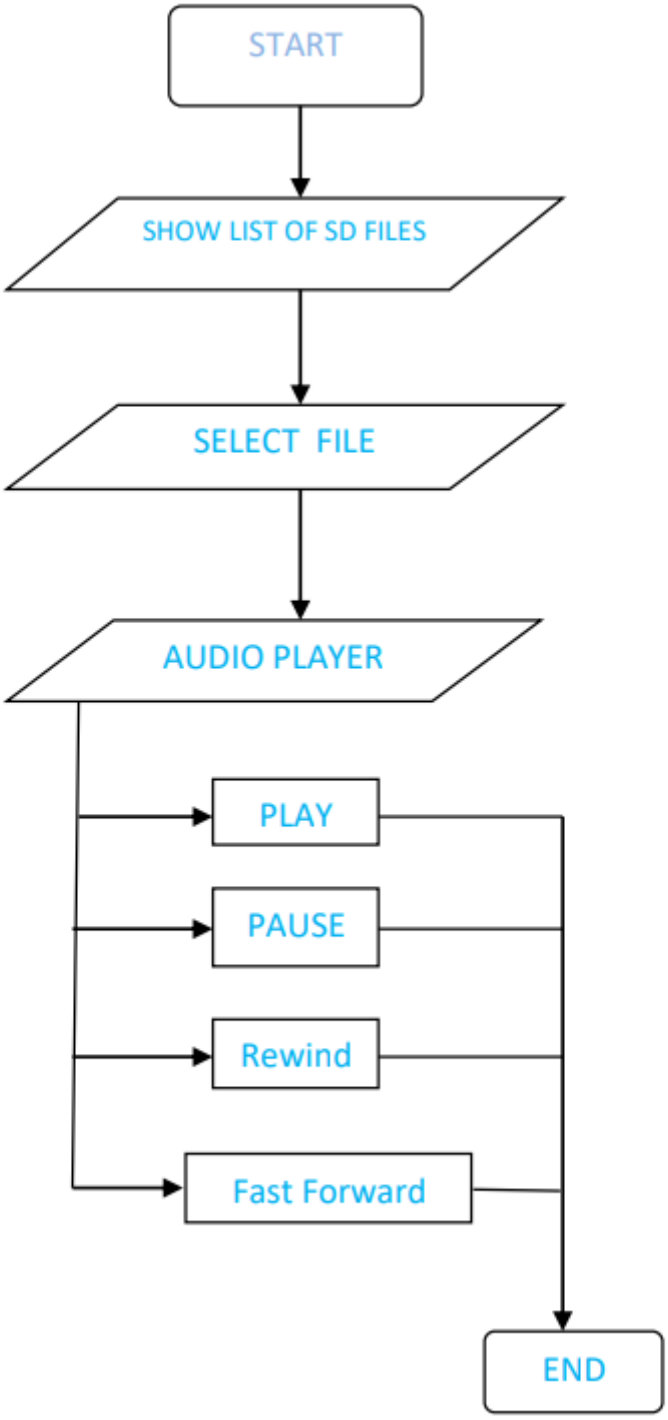
Function design of play and Next/Move Previous music

i. When need to use the player to play appropriate music, click the play button to realize the function.

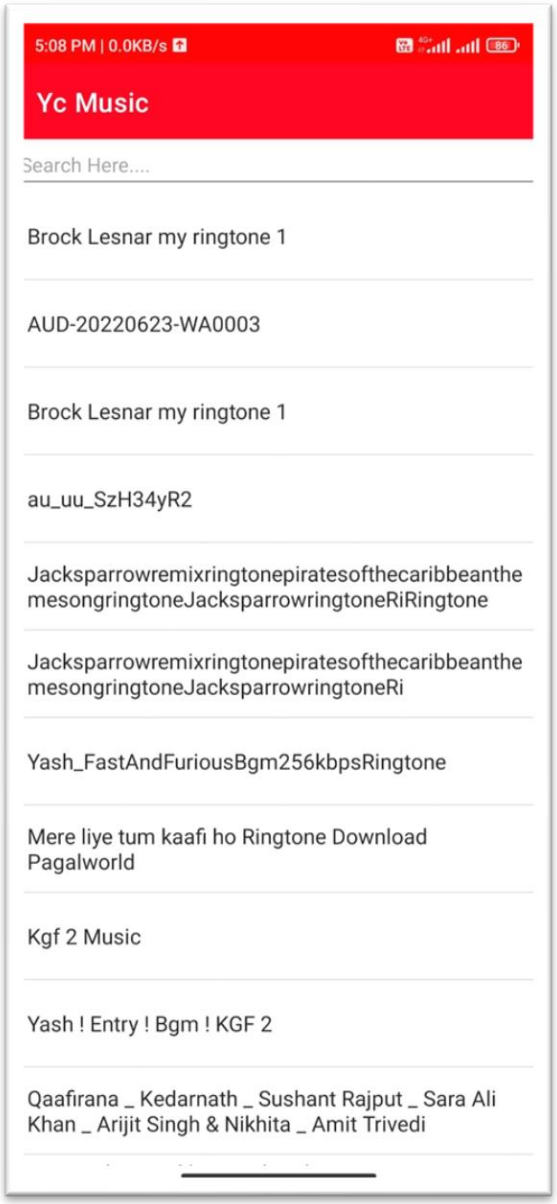
ii. When need to use the player to switch to the previous song, click on "Move Previous music" button to realize the function.

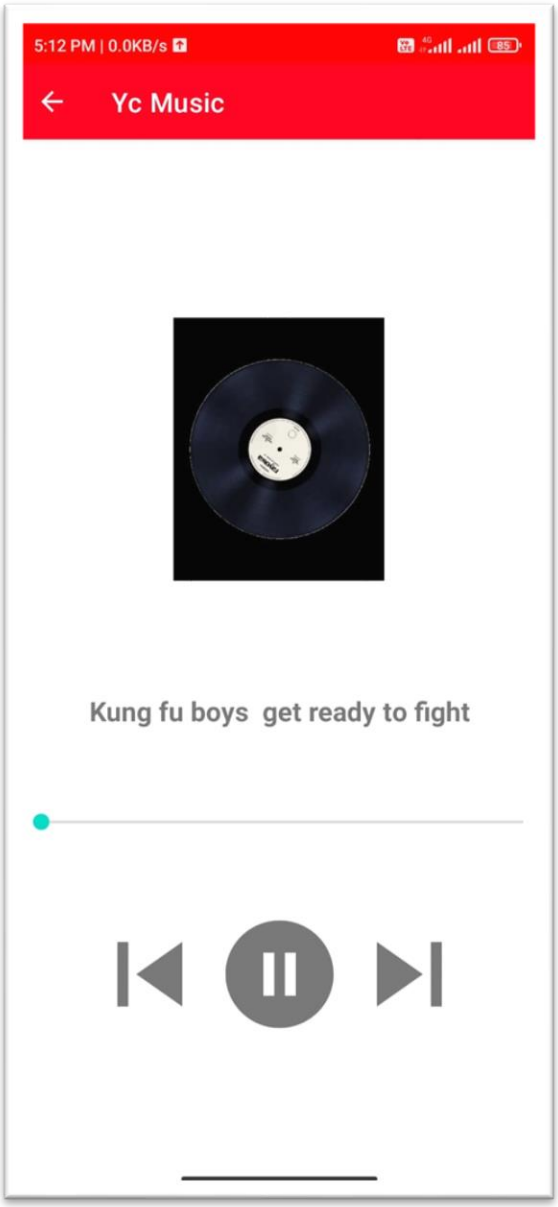
iii. When need to use the player to play the next song, click on "the next music" button to realize the function.

FLOWCHART


START
SHOW LIST OF SD FILES
SELECT FILE
AUDIO PLAYER
PLAY
PAUSE
Rewind
Fast Forward
END

APPLICATION UI





SOURCE CODE

mainActivity.java

```
package com.example.ycmusic;

import androidx.appcompat.app.AppCompatActivity;
import android.Manifest;
import android.content.Intent;
import android.os.Bundle;
import android.os.Environment;
import android.text.Editable;
import android.text.TextWatcher;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.Toast;

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 {

    EditText editText;
    ListView listView;
    ArrayList<File> mySongs;
    ArrayAdapter<String> adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        //ListView listView;//Declaration Part
        setContentView(R.layout.activity_main);
        listView = findViewById(R.id.listView);
```

```
        editText=findViewById(R.id.etSearch);


        Dexter.withContext(this)
                .withPermission(Manifest.permission.READ_EXTERNAL_STORAGE)
                .withListener(new PermissionListener() {
                    @Override
                    public void onPermissionGranted(PermissionGrantedResponse
permissionGrantedResponse) {

                        //Toast.makeText(MainActivity.this, "Runtime Permission
Given..", Toast.LENGTH_SHORT).show();
                        mySongs =
fetchSongs(Environment.getExternalStorageDirectory());//Store All Songs that we
fetch to a array list
                        String[] items = new String[mySongs.size()];//create
array to names of songs


                        for (int i = 0; i < mySongs.size(); i++) {
                            items[i] = mySongs.get(i).getName().replace(".mp3",
"");//replace name ends with mp3 by the only Name
                        }

                         adapter = new ArrayAdapter<>(MainActivity.this,
android.R.layout.simple_selectable_list_item, items); //to  display
                        listView.setAdapter(adapter);

                        listView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
                            @Override
                            public void onItemClick(AdapterView<?> parent, View
view, int position, long id) {
                                Intent intent = new Intent(MainActivity.this,
PlaySong.class);
                                String currentSong =
listView.getItemAtPosition(position).toString();
                                intent.putExtra("songList", mySongs);
                                intent.putExtra("currentSong", currentSong);
                                intent.putExtra("position", position);
                                startActivity(intent);
                            }
                        });
                        //search logic
                        editText.addTextChangedListener(new TextWatcher() {
```

```
                            @Override
                            public void beforeTextChanged(CharSequence
charSequence, int i, int i1, int i2) {

                            }

                            @Override
                            public void onTextChanged(CharSequence charSequence,
int i, int i1, int i2) {
                                adapter.getFilter().filter(charSequence);
                            }

                            @Override
                            public void afterTextChanged(Editable editable) {

                            }
                        });
                        //
                    }

                    @Override
                    public void onPermissionDenied(PermissionDeniedResponse
permissionDeniedResponse) {
                        Toast.makeText(MainActivity.this, "Please Give Permission
To Use Our App..", Toast.LENGTH_SHORT).show();
                    }

                    @Override
                    public void
onPermissionRationaleShouldBeShown(PermissionRequest permissionRequest,
PermissionToken permissionToken) {
                        permissionToken.continuePermissionRequest();//ask
permission again if not given in past time
                    }
                })

                .check();

    }
    public ArrayList<File> fetchSongs(File file){// fetch The all songs in the
Files
        ArrayList<File> arrayList=new ArrayList<>();
        File [] songs=file.listFiles();//list all files present in that directory
        //Recursive implementation
        //Take All Files Add to array list
```

```
        if (songs!=null){
          //  Toast.makeText(this, "Loading....", Toast.LENGTH_SHORT).show();
                for(File myFile: songs){// songs which we fetch give me to in
directory
                    if (!myFile.isHidden() && myFile.isDirectory()){//hiden nahi
he and Directory he
                        arrayList.addAll(fetchSongs(myFile));// add all songs
from directory to Array list
                    }
                    else {
                        if(myFile.getName().endsWith(".mp3") &&
!myFile.getName().startsWith(".")){//all file have of type mp3 amd Not appliction
sonngs
                            arrayList.add(myFile);
                        }
                    }
                }
        }
         return arrayList;
    }
```

PlaySong.java

```
package com.example.ycmusic;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.SeekBar;
import android.widget.TextView;

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

public class PlaySong extends AppCompatActivity {
```

```
    //Control O for override
    @Override
    protected void onDestroy() {//for destroy the activity on back button//stop
media player
        super.onDestroy();
        mediaPlayer.stop();
        mediaPlayer.release();
        updateSeek.interrupt();//Stop the Thread Not chalta raha hamesha
    }

    TextView textView;
    ImageView previous,play,next;
    ArrayList<File> songs;//array list to store a song//Array list coming from
Intent For that one
    MediaPlayer mediaPlayer;
    String textContent;//for song name
    int position;
    SeekBar seekBar;

    Thread updateSeek;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_play_song);

        textView=findViewById(R.id.textView);
        previous=findViewById(R.id.previous);
        play=findViewById(R.id.play);
        next=findViewById(R.id.next);
        seekBar=findViewById(R.id.seekBar);

        Intent intent=getIntent();//give me a intent
        Bundle bundle=intent.getExtras();//get the bundle object
        songs=(ArrayList)bundle.getParcelableArrayList("songList");//taking the
passed array list
        textContent= intent.getStringExtra("currentSong");//taking the name of
song and store in String
        textView.setText(textContent);//setting name

        textView.setSelected(true);//for horizontally scrolling the view LOOK
WELL

        position=intent.getIntExtra("position",0);//position of a song
```

```
        Uri uri= Uri.parse(songs.get(position).toString());//actual location of
song we have to Play

        mediaPlayer=MediaPlayer.create(this,uri);//Giving url of song which we
want to play
        mediaPlayer.start();

        seekBar.setMax(mediaPlayer.getDuration());//set maximum val of seek bar


        //seek BAR
        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());//when user touch the
seek bar
            }
        });

        updateSeek =new Thread(){
            @Override
            public void run() {
                //seekBar ko Update karte jao karte jao
                int currentPosition=0;
                try {
                    while (currentPosition < mediaPlayer.getDuration()){//update
time to if Piche chut jaye toh
                        currentPosition=mediaPlayer.getCurrentPosition();//update
seek propperly to seek Well updated
                        seekBar.setProgress(currentPosition);//set seek bar
                        sleep(800);//sleep for some millis for low use Resources
                    }
                }
                catch (Exception e){
```

```
                    e.printStackTrace();//Traditional way to catch exception in
java
                }

            }
        };
        updateSeek.start();
        //Butoons working setting
        //play
        play.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if (mediaPlayer.isPlaying()){
                    play.setImageResource(R.drawable.play);
                    mediaPlayer.pause();
                }
                else {
                    play.setImageResource(R.drawable.pause);
                    mediaPlayer.start();
                }
            }
        });
        //previous
        previous.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                mediaPlayer.stop();
                mediaPlayer.release();//stop the song first
                if (position != 0){
                    position=position-1;//for previous song
                }
                else{
                    position=songs.size() - 1;//if 1st song playing and click on
previous the play last one in the list
                }
                //After changing start the song at that postion
                Uri uri= Uri.parse(songs.get(position).toString());//actual
location of song we have to Play
                mediaPlayer=MediaPlayer.create(getApplicationContext(),uri);//Giv
ing url of song which we want to play
                mediaPlayer.start();
                play.setImageResource(R.drawable.pause);//set image after the
next or previous LOOK WELL
                seekBar.setMax(mediaPlayer.getDuration());//set maximum val of
seek bar
```

```
                //To change the name in TextView
                textContent=songs.get(position).getName();//.toString();
                textView.setText(textContent);

            }
        });
        //next
        next.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                mediaPlayer.stop();
                mediaPlayer.release();//stop the song first
                if (position != songs.size() - 1){
                    position=position+1;//for previous song
                }
                else{
                    position=0;//if 1st song playing and click on previous the
play last one in the list
                }
                //After changing start the song at that postion
                Uri uri= Uri.parse(songs.get(position).toString());//actual
location of song we have to Play
                mediaPlayer=MediaPlayer.create(getApplicationContext(),uri);//Giv
ing url of song which we want to play
                mediaPlayer.start();
                play.setImageResource(R.drawable.pause);//set image after the
next or previous LOOK WELL
                seekBar.setMax(mediaPlayer.getDuration());//set maximum val of
seek bar
               //To change the name in TextView
                textContent=songs.get(position).getName();//.toString();
                textView.setText(textContent);
            }
        });


    }
}
```

Main_Activity.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=".MainActivity">

    <EditText
        android:id="@+id/etSearch"
        android:layout_width="match_parent"
        android:layout_height="40dp"
        android:autofillHints=""
        android:ems="10"
        android:hint="@string/search_here"
        android:textSize="15sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ListView
        android:id="@+id/listView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginStart="1dp"
        android:layout_marginEnd="1dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/etSearch" />


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

CONCLUSION

Through the development of music player on Android platform, we get a clear understanding of overall process of the system. The core part of the music player is mainly composed of main interface, file browsing and song listing, Grasping the development of the music player has had the preliminary scale small features.

Music player system realized the basic function of player: play, pause, rewind and fastforward a, volume adjustment is performed through the Android System Itself, play mode, song search, seekbar, This development implicated the popular mobile terminal development technology. This is the combination management of Java language in the open source mobile platform based on Linux system configuration file. The system realized the music player programming. This design of music player based on Android system requires elaborate design of the music player framework, by adopting ANDROID STUDIO 3.1.2 + Java language as technical support of this system, with the Android plug-in tools, and combination of Latest Android SDK version lead to the comprehensive and smoothly design and development of the mobile terminal

OTHER REQUIREMENTS

- Maintain ability The design will be updated based on any changes, which are done during coding stage to maintain proper trace ability.
- Availability Available for minimum API level 22 (Android Lollipop 5.1)

REFERENCES

i. Various open source materials from Internet.

ii. Training notes.

iii. Discussion among the group and with guide.

iv. Some requirements are gathered through various books from library