背景音乐 Service 实现

完成人：林燕芝

学号：20152100089

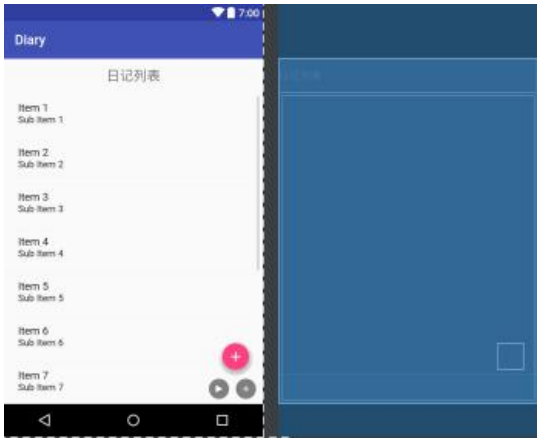
完成时间：2017.12.14

一、软件内容简介

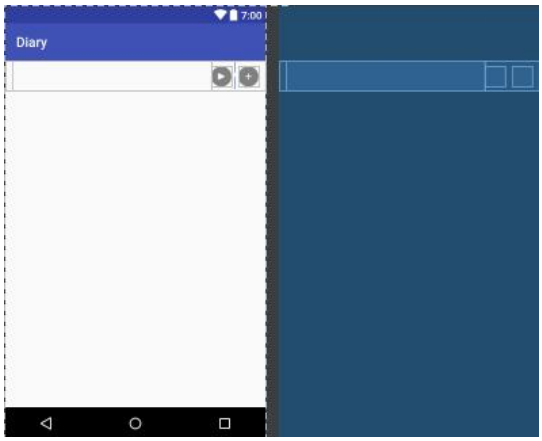
本次项目是在日记本的基础上添加播放背景音乐的功能，该功能通过 Service 实现，用户可以直接点击播放软件自带的音乐，也可以自定义添加音乐，调用系统文件管理器，选择音乐播放。

二、界面设计

本次项目的界面设计比较简单，直接在日记本的主界面添加控制音乐播放的菜单栏：



该菜单栏是一个独立的布局，可以直接 include 到各个界面，该菜单栏包含三个控件：一个用于显示播放的音乐名称的 TextView，一个用于控制音乐播放和暂停的按钮和一个用于用户自定义添加播放音乐的添加按钮：



三、代码设计

1.创建一个 Service 类，命名为 MusicService，该类用于实现音乐的播放和暂停功能，根据传入的 path 是否为空判断是播放系统自带音乐还是播放用户自定义添加的本地音乐，如果是前者则 create(getApplicationContext(),R.raw.kiss_the_rain)，后者则 setDataSource(path)：

```
public class MusicService extends Service {
    private static final String TAG="MusicService";
    public MediaPlayer mMediaPlayer;
    private String prepath="";
    private boolean xitong=false;

    class MyBinder extends Binder{
        public void play(String path){
            //播放系统自带的背景音乐后自定义添加播放音乐
            if(!path.equals("")&&xitong){
                mMediaPlayer.release();
                mMediaPlayer=null;
            }
            try{
                if(mMediaPlayer==null){
                    prepath=path;
                    //path 为空时
                    if(path.equals("")){
                mMediaPlayer=MediaPlayer.create(getApplicationContext(),
                                R.raw.kiss_the_rain);
                        mMediaPlayer.start();
                        xitong=true;
                    }
                    else{
                        xitong=false;
```

```
                    //path 有效
                    mMediaPlayer=new MediaPlayer();
                mMediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);
                    mMediaPlayer.setDataSource(path);
                    mMediaPlayer.prepare();
                    mMediaPlayer.setOnPreparedListener(new
MediaPlayer.OnPreparedListener() {
                        @Override
                        public void onPrepared(MediaPlayer mp) {
                            mMediaPlayer.start();
                        }
                    });
                }
            }
            else{
                if(path.equals("")){
                    xitong=true;
                    mMediaPlayer.release();
                    mMediaPlayer=MediaPlayer.create(getApplicationContext(),
                                    R.raw.kiss_the_rain);
                }
                if(prepath.equals(path)){
                    int position=getCurrentProgress();
                    mMediaPlayer.seekTo(position);
                    try{
                        mMediaPlayer.prepare();
                    }catch (Exception e){
                        e.printStackTrace();
                    }
                    mMediaPlayer.start();
```

```
                    }
                    else{
                        prepath=path;
                        //重新选择音乐
                        mMediaPlayer.reset();
                        mMediaPlayer.setDataSource(path);
                        mMediaPlayer.prepare();
                        mMediaPlayer.setOnPreparedListener(new
MediaPlayer.OnPreparedListener() {
                            @Override
                            public void onPrepared(MediaPlayer mp) {
                                mMediaPlayer.start();
                            }
                        });
                    }
                }
            }catch (Exception e){
                e.printStackTrace();
            }
        }

        public void pause(){
            if(mMediaPlayer!=null&&mMediaPlayer.isPlaying()){
                mMediaPlayer.pause();
            }
            else if(mMediaPlayer!=null&&(!mMediaPlayer.isPlaying())){
                mMediaPlayer.start();
            }
        }
    }
```

```
    public void onCreate(){
        super.onCreate();
    }
    //获取当前进度
    public int getCurrentProgress(){
        if(mMediaPlayer!=null&mMediaPlayer.isPlaying()){
            return mMediaPlayer.getCurrentPosition();
        }
        else if(mMediaPlayer!=null&(!mMediaPlayer.isPlaying())){
            return mMediaPlayer.getCurrentPosition();
        }
        return 0;
    }
    //退出系统时释放资源
    public void onDestroy(){
        if(mMediaPlayer!=null){
            mMediaPlayer.stop();
            mMediaPlayer.release();
            mMediaPlayer=null;
        }
        super.onDestroy();
    }
    @Override
    public IBinder onBind(Intent intent) {
        // TODO: Return the communication channel to the service.
        //throw new UnsupportedOperationException("Not yet implemented");
        return new MyBinder();
    }
}
```

2.在 Activity 中，声明菜单栏控件并初始化：

```
private TextView tvmusicname;
private ImageButton ibplay;
private ImageButton ibaddmus;
private myConn conn;
private String musicpath="";
private MusicService.MyBinder binder;
private boolean isplay=false;
//绑定服务的intent
Intent MusicServiceIntent;
```

```
tvmusicname=(TextView)findViewById(R.id.tv_name);
ibplay=(ImageButton)findViewById(R.id.ib_play);
ibaddmus=(ImageButton)findViewById(R.id.ib_addmusic);
conn=new myConn();
MusicServiceIntent=new Intent(this,MusicService.class);
```

3.在 Activity 中，声明 Service 和 Activity 的连接类：

```
private class myConn implements ServiceConnection{
    public void onServiceConnected(ComponentName name, IBinder service){
        binder=(MusicService.MyBinder) service;
    }

    public void onServiceDisconnected(ComponentName name){

    }
}
```

4.由于需要播放 Android 系统本地音乐，因此需要获取存储读写权限才能播放，因此在 AndroidManifest.xml 中添加权限声明：

<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>

<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

在 onCreate 中，动态申请该权限：

```
if (ContextCompat.checkSelfPermission(MainActivity.this, Manifest.
        permission.WRITE_EXTERNAL_STORAGE) != PackageManager.
        PERMISSION_GRANTED) {
    ActivityCompat.requestPermissions(MainActivity.this, new String[]{
            Manifest.permission.WRITE_EXTERNAL_STORAGE
    }, 1);
} else {
    //够了就设置路径等，准备播放
    bindService(MusicServiceIntent,conn,BIND_AUTO_CREATE);
}
```

运行时权限回调：

```
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[]
        permissions, @NonNull int[] grantResults) {
    switch (requestCode) {
        case 1:
            if (grantResults.length > 0 && grantResults[0] ==
                    PackageManager.PERMISSION_GRANTED) {
                bindService(MusicServiceIntent,conn,BIND_AUTO_CREATE);
            } else {
                Toast.makeText(this, "权限不够获取不到音乐，程序将退出",
                        Toast.LENGTH_SHORT).show();
                finish();
            }
            break;
        default:
            break;
    }
}
```

5.自定义添加音乐 button 事件，调用 Android 系统的文件管理器，设置文件类型为 audio，打开文件管理器选本地音乐：

```
ibaddmus.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent=new Intent(Intent.ACTION_GET_CONTENT);
        //系统调用Action属性
        intent.setType("audio/*");
        //设置文件类型
        intent.addCategory(Intent.CATEGORY_OPENABLE);
        // 添加Category属性
        try{
            startActivityForResult(intent,2);
        }catch(Exception e){
            Toast.makeText(MainActivity.this, "没有正确打开文件管理器",
                    Toast.LENGTH_SHORT).show();
        }

    }
});
```

6.调用系统文件管理器返回主界面事件：

@Override

protected void onActivityResult(int requestCode, int resultCode,

Intent data) {

super.onActivityResult(requestCode, resultCode, data);

if(data!=null){

if(requestCode==2){

if(resultCode== Activity.RESULT_OK){

Uri uri = data.getData();

//得到 uri，后面就是将 uri 转化成 file 的过程

```
                    //得到的 uri 的路径并不是真实的路径，因此需要进一步转换
                    String url=uri.getPath();
                    musicpath=url;
                    File SDpath= Environment.getExternalStorageDirectory();
                    String[] datastr=musicpath.split("/");
                    String realpath="";
                    for(int i=2;i<datastr.length;i++){
                        realpath=realpath+"/"+datastr[i];
                    }
                    File file=new File(SDpath,realpath);
                    String path=file.getAbsolutePath();
                    String musicname=datastr[datastr.length-1];
                    tvmusicname.setText(musicname);
                    //得到有效的路径，实现音乐播放
                    if(file.exists()&&file.length()>0) {
                        binder.play(path);
                        ibplay.setImageResource(R.mipmap.pause);
                        isplay = true;
                    }
                }
            }
        }
    }
```

7.播放暂停按钮的事件：判断正在播放标志 isplay 是否为 true，为 true 则调用 binder.pause()，并设置正在播放标志 isplay 为 true，为 false 则调用 binder.play(path)，并设置正在播放标志 isplay 为 true，同时更改图标；当 musicpath 为空时，直接调用 binder.play(path)；当 musicpath 不为空，则利用 musicpath 获取真实路径，并判断该文件是否存在，实现音乐播放：

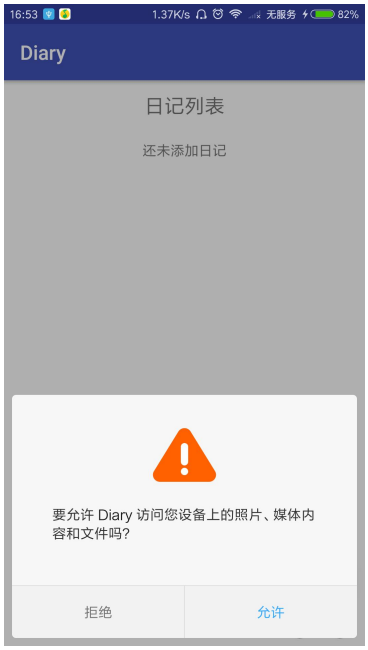
```
ibplay.setOnClickListener(new View.OnClickListener() {
            @Override
```

```
public void onClick(View v) {
        String path;
        File file=null;
        //musicpath 不为空时，获取想要获取的音乐
        if(!musicpath.equals("")){
            File SDpath= Environment.getExternalStorageDirectory();
            String[] datastr=musicpath.split("/");
            String realpath="";
            for(int i=2;i<datastr.length;i++){
                realpath=realpath+"/"+datastr[i];
            }
            file=new File(SDpath,realpath);
            path=file.getAbsolutePath();
            String musicname=datastr[datastr.length-1];
            tvmusicname.setText(musicname);
        }
        else {
            //为空是设置 path 为空
            tvmusicname.setText("kiss the rain.mp3");
            path="";
        }
        if(musicpath.equals("")||(file.exists()&&file.length()>0)){
            if(!isplay){
                binder.play(path);
                ibplay.setImageResource(R.mipmap.pause);
                isplay=true;
            }
            else{
                binder.pause();
                ibplay.setImageResource(R.mipmap.play);
```

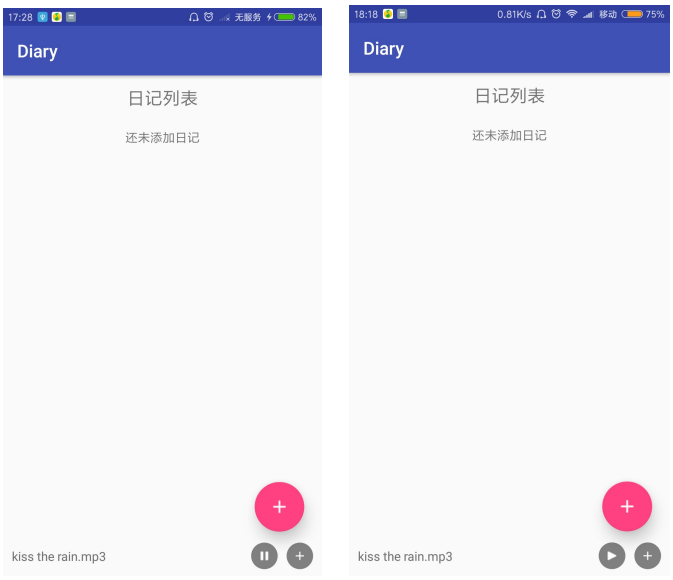
```
                    isplay=false;
                }
            }
            else{
                //文件流不存在，输出错误提醒
                Toast.makeText(MainActivity.this," 找 不 到 音 乐 文 件 ",Toast.LENGTH_SHORT).show();
            }
        }
    });
```

四、软件操作流程

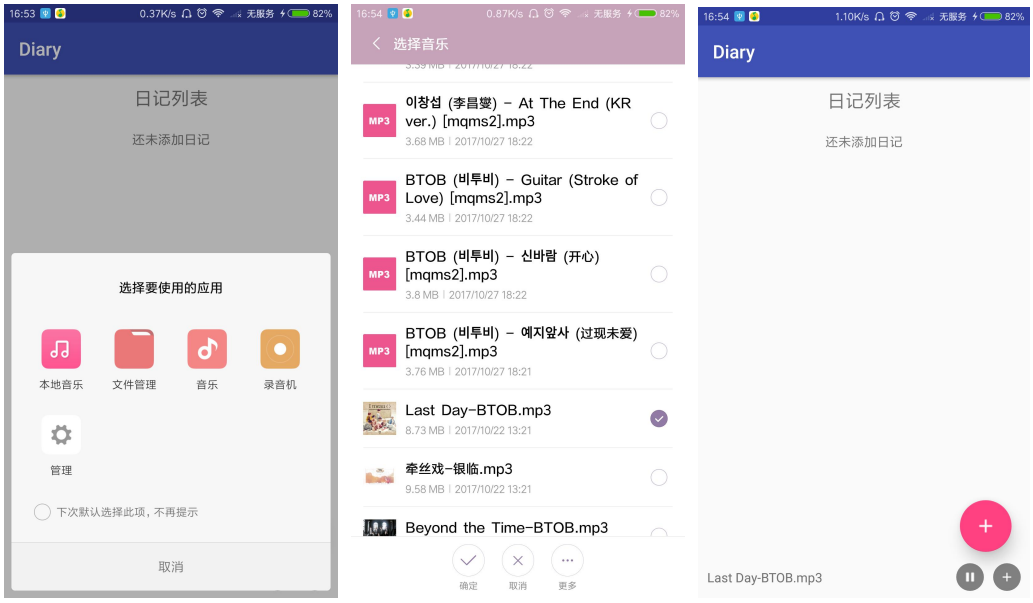
1.打开日记本 app，系统询问是否允许获取存储读写权限（不允许该权限则无法播放用户自定义添加的音乐）：



2.直接点击播放按钮，直接播放系统自带的音乐，点击暂停图标停止播放：



3.点击添加按钮，调用系统自带的文件管理器，选择想要添加的音乐，选择成功返回到程序则开始播放：



五、难点和解决方案

难点：调用 Android 系统的文件管理器，获取有效路径。

解决方案：对比系统的真实路径和 uri 得到的路径，去掉 uri 得到的路径前面无效部分，只使用后半部分，添加到 Environment.getExternalStorageDirectory()后，得到有效的路径。

六、不足之处

只能添加一首音乐，功能比较简单。

七、今后设想

进一步完成背景音乐播放功能，实现批量添加音乐；添加进度条，查看调节播放进度的功能；记录上次的歌曲列表和播放歌曲的功能。