**实 验 报 告**

**课程名称 移动应用系统**

**实验项目 移动平台下多媒体的应用**

**实验仪器 计算机一台、Android手机一部**

**系 别 计算机学院**

**专 业 计算机科学与技术**

**班级/学号**  计科1606 / 2016010311

**学生姓名**  耿瑞

**实验日期**  2018年12月28日

**成 绩** \_\_\_\_\_\_\_\_\_\_

**指导教师** \_\_\_\_ **郝保水**\_\_\_\_ \_\_

1 实验目的

（1）了解Android Service的基本概念；

（2）理解并掌握Service使用方法;

（3）掌握MediaPlayer的使用方法；

2 实验要求

（1）课前预习实验内容，并查找相关资料。

（2）按照实验步骤完成各个相关内容。

（3）撰写实验报告。

A） 实验报告格式必须符合学校要求（例如必须采用学校规定的实验封面）；

B） 写出实验详细步骤，包括主要采用的技术方案、相关分析和核心代码。注意：不要简单地近包括截屏和代码，完整代码可以作为附录放在实验报告结尾；

C） 总结实验中遇到的问题、分析和解决方法。

D） 写出心得体会与收获等。

3 实验步骤

主要分为几个大步骤进行实现。

（1）需求分析，调研播放器具有哪些功能，确定界面。

（2）系统设计，对系统进行模块划分，确定技术方案。

（3）编码，编写代码实现各项功能。

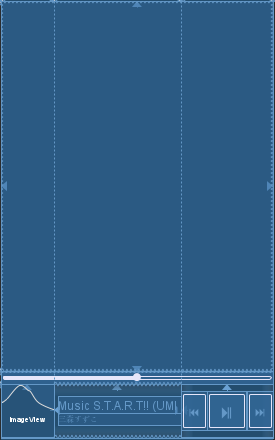
（4）测试。

3.1 需求分析

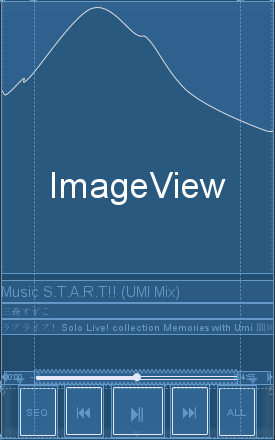
（1）主要功能

* 用户选择音乐文件
* 列表播放
* 多种播放模式：单次播放、单曲循环播放、列表顺序循环播放、列表随机播放

（2）界面设计



主界面（播放列表）



详细播放器界面

3.2 系统设计

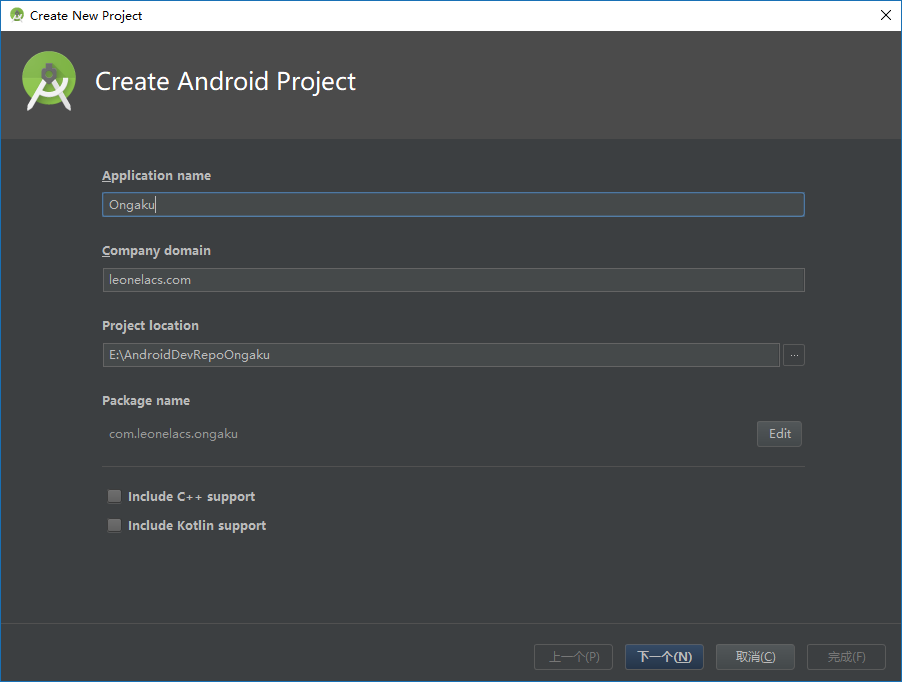
播放器包括以下模块：

* 主界面（播放列表）模块
* 详细播放器界面模块
* 音乐播放服务模块

3.3 项目实施

主要步骤：

（1）创建工程



在Android Studio中创建Android项目。

应用名称：Ongaku

域名：leonelacs.com

包名：com.leonelacs.ongaku

最低SDK版本API 24: Android 7.0

（2）编写各界面布局、菜单等资源文件

主界面（播放列表），使用ConstraintLayout和LinearLayout实现，包含：

* 用于显示播放列表的RecyclerView：**OngakuRecycler**；
* 用于显示和控制播放进度的SeekBar：**MainSeekBar**；
* 用于显示当前音乐专辑图片的ImageView **MainAlbumCover**
* 用于显示当前音乐名称的TextView **MainTitle**
* 用于显示当前音乐艺术家的TextView **MainArtist**
* 用于播放上一首的Button：**MainButtonPrevious**
* 用于播放/暂停的Button：**MainButtonPlay**
* 用于播放下一首的Button：**MainButtonNext**

OngakuRecycler响应Item的点击事件，当点击其中的项目时，播放该项目的音乐。

MainSeekBar响应改变事件，当改变该控件的进度时，改变正在播放的音乐的播放进度。

MainAlbumCover响应点击事件，当点击该控件时，打开详细播放器界面。

MainButtonPrevious响应点击事件，当点击该控件时，播放上一首音乐。

MainButtonPlay响应点击事件，当点击该控件时，继续播放/暂停播放音乐。

MainButtonNext响应点击事件，当点击该控件时，播放下一首音乐。



详细播放器界面，使用ConstraintLayout和LinearLayout实现，包含：

* 用于显示当前音乐专辑图片的ImageView **DetailAlbumCover**
* 用于显示当前音乐名称的TextView **DetailTitle**
* 用于显示当前音乐艺术家的TextView **DetailArtist**
* 用于显示当前音乐专辑名称的TextView **DetailAlbumName**
* 用于显示和控制播放进度的SeekBar：**DetailSeekBar**
* 用于显示当前播放进度时长的TextView：**DetailCurrent**
* 用于显示当前播放音乐总时长的TextView：**DetailFull**
* 用于切换顺序/随机播放的Button：**DetailButtonShaffuru**
* 用于播放上一首的Button：**DetailButtonPrevious**
* 用于播放/暂停的Button：**DetailButtonPlay**
* 用于播放下一首的Button：**DetailButtonNext**
* 用于切换循环播放模式的Button：**DetailButtonRipiito**

DetailSeekBar响应改变事件，当改变该控件的进度时，改变正在播放的音乐的播放进度。

DetailButtonShaffuru响应点击事件，当点击该控件时，播放切换顺序/随机播放。

DetailButtonPrevious响应点击事件，当点击该控件时，播放上一首音乐。

DetailButtonPlay响应点击事件，当点击该控件时，继续播放/暂停播放音乐。

DetailButtonNext响应点击事件，当点击该控件时，播放下一首音乐。

DetailButtonRipiito响应点击事件，当点击该控件时，切换循环播放模式。



详细播放器界面中的正方形专辑图片由自己编写的继承自RelativeLayout的SquareRelativeLayout实现，重写了onMeasure方法。

重写的onMeasure方法：

protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {  
 super.onMeasure(widthMeasureSpec, widthMeasureSpec);  
}

详细播放器界面背景为高斯模糊后的当前播放音乐的专辑图片（若当前播放音乐无专辑图片则显示蓝色纯色背景），高斯模糊由RenderScript及ScriptIntrinsicBlur实现

在DetailActivity中有以下两个方法完成图片的高斯模糊：

* 模糊图片（半径25） Bitmap blurBitmap(Bitmap bmp)
* 重复三次模糊图片 Bitmap blur3Times(Bitmap bmp)

在DetailActivity中还有String toTimeFormat(int time\_mm)方法将毫秒时长转换为MM:ss格式的字符串。

音乐播放服务HousouService

通过读取Application中的音乐对象列表获得音乐路径，由MediaPlayer实例mediaPlayer进行播放。

服务提供以下方法：

* + 获取播放进度 public int getHousouProgress()
  + 获取当前播放音乐长度 public int getHousouDuration()
  + 设置播放进度 public void setHousouProgress(int time)
  + 获取循环模式 public RipiitoMode getRepeat()
  + 获取切歌模式 public ShaffuruMode getShuffle()
  + 变更循环模式 public RipiitoMode changeRepeat()
  + 变更切歌模式 public ShaffuruMode changeShuffle()
  + 播放（继续播放） public int play()
  + 暂停播放 public void pause()
  + 播放上一首 public int previous()
  + 播放下一首 public int next()
  + 播放所选音乐 public void select(int index)
  + 获取播放状态 public boolean getIsPlaying()
  + 获取正在播放音乐索引 public int getHIndex()

mediaPlayer响应播放完成事件，当播放完成时，根据循环模式和切歌模式进行动作。

Application OngakuApp

在Application创建时，从系统MediaStore获得所有音乐的信息，形成全局的音乐信息对象列表List<Ongaku> globalOngakus。

主界面、详细播放器页面和播放服务均从globalOngakus获取音乐信息。

获取音乐信息对象列表：

public List<Ongaku> getOngakuFromPhone() {  
 ContentResolver contentResolver = getContentResolver();  
 Cursor cursor = contentResolver.query(MediaStore.Audio.Media.*EXTERNAL\_CONTENT\_URI*,null,null,null,MediaStore.Audio.Media.*DEFAULT\_SORT\_ORDER*);  
 List<Ongaku> myuujikkuFiles = new ArrayList<>();  
 if (cursor.moveToFirst()) {  
 do {  
 Ongaku m = new Ongaku();  
 //中间省略  
 if (ongakuDesu != 0 && duration / (500 \* 60) >= 1) {  
 //中间省略  
 myuujikkuFiles.add(m);  
 }  
 }  
 while (cursor.moveToNext());  
 }  
 cursor.close();  
 return myuujikkuFiles;  
}

OngakuAdapter中声明并实现了OnItemClickListener接口，以便于在Activity中处理OngakuAdapter的Item的点击事件。

private OnItemClickListener onItemClickListener;  
  
public interface OnItemClickListener {  
 void onItemClick(View view, int position, Ongaku ongaku);  
}  
  
private class MyOnClickListener implements View.OnClickListener {  
 private int position;  
 private Ongaku ongaku;  
  
 public MyOnClickListener(int position, Ongaku ongaku) {  
 this.position = position;  
 this.ongaku = ongaku;  
 }  
  
 @Override  
 public void onClick(View view) {  
 onItemClickListener.onItemClick(view, position, ongaku);  
 }  
}  
  
public void setOnItemClickListener(OnItemClickListener onItemClickListener) {  
 this.onItemClickListener = onItemClickListener;  
}

3.4 测试

（1）点击列表项目播放所选音乐

在主界面点击OngakuRecycler中的项目，播放器开始播放所选音乐

例：点击「僕たちはひとつの光」，播放器开始播放该歌曲



（2）进入详细播放器界面

在主界面点击左下角专辑图片，进入详细播放器界面。

例：点击左下角专辑图片，进入详细播放器界面



（3）拖动进度条改变播放进度

在主界面或详细播放器界面拖动进度条，改变播放进度。

例：在详细播放器界面拖动进度条，改变播放进度



（3）点击⏮/⏭按钮播放上一首/下一首音乐

在主界面或详细播放器界面点击⏮/⏭按钮播放上一首/下一首音乐。

例：在主界面点击⏭按钮，播放「僕らのLIVE 君とのLIFE」



4 实验中遇到的问题、分析和解决方法

（1）全局音乐信息列表

音乐信息列表原本写在MainActivity中（ongakuList），但是在应用的编写过程中发现，在DetailActivity中也会用到列表中的内容，需要传递多个副本，会大量占用系统资源。经过查找资料，发现可以将列表定义在Application中，以实现全局调用。

（2）Bitmap对象占用大量内存空间

在保存专辑图片时占用了大量的内存空间，最开始甚至高达1.5GB，也造成了RecyclerView拖动时的卡顿。经查，Bitmap对象占据的内存空间会是其编码体积的十余倍，于是在保存前对Bitmap对象进行压缩，最终将内存空间占用降至200MB左右。

（3）希望在Activity中获取RecyclerView被点击项目的索引

OngakuAdapter中声明并实现了OnItemClickListener接口，然后就可以在Activity中获得被点击项目的索引并处理的点击事件。

（4）希望获得与屏幕同宽的正方形ImageView

编写了继承自RelativeLayout的SquareRelativeLayout实现，重写了onMeasure方法，将widthMeasureSpec既作为宽也作为高传给父类的onMeasure方法。

5 心得体会与收获

通过这次实验，我完成了一个音乐播放器Android应用的开发。了解了Android Service的基本概念；理解并掌握了Service使用方法；掌握了MediaPlayer的使用方法。

这次实验的音乐播放器的外观与功能是以索尼手机自带的音乐播放器为范本设计的。通过查找资料，在Service中实现了音乐的播放功能；在Application中实现了全局可用的音乐信息列表；声明并实现了OngakuAdapter的OnItemClickListener接口，以在Activity中处理点击事件；编写了正方形相对布局类，将ImageView置于其中，实现与屏幕同宽的正方形图片框。除此之外，还通过Android的库方法实现了图片的高斯模糊，并以此作为详细播放器界面的背景。

与索尼手机自带音乐播放器的外观对比

自己编写的音乐播放器外观：

索尼手机自带音乐播放器外观:

6 附录

代码仓库：<https://github.com/leonelacs/Ongaku>

程序完整代码：

**com.leonelacs.ongaku**

Ongaku.java

package com.leonelacs.ongaku;  
  
import android.graphics.Bitmap;  
  
public class Ongaku {  
 private long id = 0;  
 private String title = "";  
 private String artist = "";  
 private String album = "";  
 private long albumId = 0;  
 private String path = "";  
 private long duration = 0;  
 private long size = 0;  
 private Bitmap cover = null;  
  
 public Ongaku() {}  
  
 public Ongaku(long id, String title, String artist, String album, long albumId, String path, long duration, long size, Bitmap cover) {  
 this.id = id;  
 this.title = title;  
 this.artist = artist;  
 this.album = album;  
 this.albumId = albumId;  
 this.path = path;  
 this.duration = duration;  
 this.size = size;  
 this.cover = cover;  
 }  
  
 public long getId() {  
 return id;  
 }  
  
 public void setId(long id) {  
 this.id = id;  
 }  
  
 public String getTitle() {  
 return title;  
 }  
  
 public void setTitle(String title) {  
 this.title = title;  
 }  
  
 public String getArtist() {  
 return artist;  
 }  
  
 public void setArtist(String artist) {  
 this.artist = artist;  
 }  
  
 public String getAlbum() {  
 return album;  
 }  
  
 public void setAlbum(String album) {  
 this.album = album;  
 }  
  
 public long getAlbumId() {  
 return albumId;  
 }  
  
 public void setAlbumId(long albumId) {  
 this.albumId = albumId;  
 }  
  
 public String getPath() {  
 return path;  
 }  
  
 public void setPath(String path) {  
 this.path = path;  
 }  
  
 public long getDuration() {  
 return duration;  
 }  
  
 public void setDuration(long duration) {  
 this.duration = duration;  
 }  
  
 public long getSize() {  
 return size;  
 }  
  
 public void setSize(long size) {  
 this.size = size;  
 }  
  
 public Bitmap getCover() {  
 return cover;  
 }  
  
 public void setCover(Bitmap cover) {  
 this.cover = cover;  
 }  
}

RipiitoMode.java

package com.leonelacs.ongaku;  
  
public enum RipiitoMode {  
 *NO\_LOOP*,  
 *SINGLE\_LOOP*,  
 *ALL\_LOOP*}

Shaffuru.java

package com.leonelacs.ongaku;  
  
public enum ShaffuruMode {  
 *ORDER*,  
 *RANDOM*}

SquareRelativeLayout.java

package com.leonelacs.ongaku;  
  
import android.content.Context;  
import android.util.AttributeSet;  
import android.widget.RelativeLayout;  
  
public class SquareRelativeLayout extends RelativeLayout {  
 public SquareRelativeLayout(Context context) {  
 super(context);  
 }  
 public SquareRelativeLayout(Context context, AttributeSet attrs) {  
 super(context, attrs);  
 }  
 public SquareRelativeLayout(Context context, AttributeSet attrs, int defStyleAttr) {  
 super(context, attrs, defStyleAttr);  
 }  
 @Override  
 protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {  
 super.onMeasure(widthMeasureSpec, widthMeasureSpec);  
 }  
}

OngakuAdapter.java

package com.leonelacs.ongaku;  
  
import android.content.Intent;  
import android.support.v7.widget.RecyclerView;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.AdapterView;  
import android.widget.ImageView;  
import android.widget.TextView;  
  
import java.util.List;  
  
public class OngakuAdapter extends RecyclerView.Adapter<OngakuAdapter.ViewHolder> {  
 private List<Ongaku> ongakus;  
 private OnItemClickListener onItemClickListener;  
  
 public interface OnItemClickListener {  
 void onItemClick(View view, int position, Ongaku ongaku);  
 }  
  
 private class MyOnClickListener implements View.OnClickListener {  
 private int position;  
 private Ongaku ongaku;  
  
 public MyOnClickListener(int position, Ongaku ongaku) {  
 this.position = position;  
 this.ongaku = ongaku;  
 }  
  
 @Override  
 public void onClick(View view) {  
 onItemClickListener.onItemClick(view, position, ongaku);  
 }  
 }  
  
 public void setOnItemClickListener(OnItemClickListener onItemClickListener) {  
 this.onItemClickListener = onItemClickListener;  
 }  
  
 static class ViewHolder extends RecyclerView.ViewHolder {  
 ImageView albumCover;  
 TextView musicTitle;  
 TextView musicArtist;  
 View searchView;  
  
 public ViewHolder(View view) {  
 super(view);  
 searchView = view;  
 albumCover = (ImageView) view.findViewById(R.id.*album\_cover*);  
 musicTitle = (TextView) view.findViewById(R.id.*music\_title*);  
 musicArtist = (TextView) view.findViewById(R.id.*music\_artist*);  
 }  
 }  
  
 public OngakuAdapter(List<Ongaku> ongakuList) {  
 ongakus = ongakuList;  
 }  
  
 @Override  
 public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {  
 View view = LayoutInflater.*from*(parent.getContext()).inflate(R.layout.*ongaku\_square*, parent, false);  
 ViewHolder holder = new ViewHolder(view);  
  
// holder.searchView.setOnClickListener(new View.OnClickListener() {  
// @Override  
// public void onClick(View view) {  
// int position = holder.getAdapterPosition();  
// Ongaku ongakuBeClicked = ongakus.get(position);  
// }  
// });  
  
 return holder;  
 }  
  
 @Override  
 public void onBindViewHolder(ViewHolder holder, int position) {  
 Ongaku ongaku = ongakus.get(position);  
 int adapterPosition = holder.getAdapterPosition();  
 if (ongaku.getCover() != null) {  
 holder.albumCover.setImageBitmap(ongaku.getCover());  
 }  
 else {  
 holder.albumCover.setImageResource(R.drawable.*default\_cover\_alpha\_400*);  
 }  
 holder.musicTitle.setText(ongaku.getTitle());  
 holder.musicArtist.setText(ongaku.getArtist());  
 if (onItemClickListener != null) {  
 holder.itemView.setOnClickListener(new MyOnClickListener(position, ongakus.get(adapterPosition)));  
 }  
 }  
  
 @Override  
 public int getItemCount() {  
 return ongakus.size();  
 }  
}

HousouService.java

package com.leonelacs.ongaku;  
  
import android.app.Service;  
import android.content.Intent;  
import android.media.MediaPlayer;  
import android.os.Binder;  
import android.os.IBinder;  
  
import java.util.List;  
  
public class HousouService extends Service {  
  
 MediaPlayer mediaPlayer = new MediaPlayer();  
 HousouBinder housouBinder;  
// List<String> ongakuPathList;  
 String hPath;  
 int hPosition;  
 int hIndex;  
 RipiitoMode repeat = RipiitoMode.*ALL\_LOOP*;  
 ShaffuruMode shuffle = ShaffuruMode.*ORDER*;  
  
 OngakuApp ongakuApp;  
  
  
 public HousouService() {  
 }  
  
 @Override  
 public IBinder onBind(Intent intent) {  
// // *TODO: Return the communication channel to the service.*// throw new UnsupportedOperationException("Not yet implemented");  
 return housouBinder;  
 }  
  
 class HousouBinder extends Binder {  
// public void receiveOngakuPathList(List<String> paths) {  
// ongakuPathList.clear();  
// ongakuPathList.addAll(paths);  
// }  
  
 public int getHousouProgress() {  
 return mediaPlayer.getCurrentPosition();  
 }  
  
 public int getHousouDuration() {  
 return mediaPlayer.getDuration();  
 }  
  
 public void setHousouProgress(int time) {  
 mediaPlayer.seekTo(time);  
 }  
  
 public RipiitoMode getRepeat() {  
 return repeat;  
 }  
  
 public ShaffuruMode getShuffle() {  
 return shuffle;  
 }  
  
 public RipiitoMode changeRepeat() {  
 if (repeat == RipiitoMode.*NO\_LOOP*) { repeat = RipiitoMode.*SINGLE\_LOOP*; }  
 else if (repeat == RipiitoMode.*SINGLE\_LOOP*) { repeat = RipiitoMode.*ALL\_LOOP*; }  
 else { repeat = RipiitoMode.*NO\_LOOP*; }  
 return repeat;  
 }  
  
 public ShaffuruMode changeShuffle() {  
 if (shuffle == ShaffuruMode.*ORDER*) { shuffle = ShaffuruMode.*RANDOM*; }  
 else { shuffle = ShaffuruMode.*ORDER*; }  
 return shuffle;  
 }  
  
 public int play() {  
 if (hPath == null) {  
 try {  
 mediaPlayer.reset();  
 hPath = ongakuApp.globalOngakus.get(0).getPath();  
 hIndex = 0;  
 mediaPlayer.setDataSource(hPath);  
 mediaPlayer.prepare();  
 mediaPlayer.start();  
 }  
 catch (Exception e) { e.printStackTrace(); }  
 }  
 else {  
 mediaPlayer.start();  
 }  
 return hIndex;  
 }  
 public void pause() { mediaPlayer.pause(); }  
  
 public int previous() {  
 if (hIndex == 0) {  
 try {  
 mediaPlayer.reset();  
 hIndex = ongakuApp.globalOngakus.size() - 1;  
 hPath = ongakuApp.globalOngakus.get(hIndex).getPath();  
 mediaPlayer.setDataSource(hPath);  
 mediaPlayer.prepare();  
 mediaPlayer.start();  
 }  
 catch (Exception e) { e.printStackTrace(); }  
 }  
 else {  
 try {  
 mediaPlayer.reset();  
 hIndex = hIndex - 1;  
 hPath = ongakuApp.globalOngakus.get(hIndex).getPath();  
 mediaPlayer.setDataSource(hPath);  
 mediaPlayer.prepare();  
 mediaPlayer.start();  
 }  
 catch (Exception e) { e.printStackTrace(); }  
 }  
 return hIndex;  
 }  
  
 public int next() {  
 if (hIndex == ongakuApp.globalOngakus.size() - 1) {  
 try {  
 mediaPlayer.reset();  
 hIndex = 0;  
 hPath = ongakuApp.globalOngakus.get(hIndex).getPath();  
 mediaPlayer.setDataSource(hPath);  
 mediaPlayer.prepare();  
 mediaPlayer.start();  
 }  
 catch (Exception e) { e.printStackTrace(); }  
 }  
 else {  
 try {  
 mediaPlayer.reset();  
 hIndex = hIndex + 1;  
 hPath = ongakuApp.globalOngakus.get(hIndex).getPath();  
 mediaPlayer.setDataSource(hPath);  
 mediaPlayer.prepare();  
 mediaPlayer.start();  
 }  
 catch (Exception e) { e.printStackTrace(); }  
 }  
 return hIndex;  
 }  
  
 public void select(int index) {  
 try {  
 mediaPlayer.reset();  
 hIndex = index;  
 hPath = ongakuApp.globalOngakus.get(hIndex).getPath();  
 mediaPlayer.setDataSource(hPath);  
 mediaPlayer.prepare();  
 mediaPlayer.start();  
 }  
 catch (Exception e) { e.printStackTrace(); }  
 }  
  
 public boolean getIsPlaying() {  
 return mediaPlayer.isPlaying();  
 }  
  
 public int getHIndex() {  
 return hIndex;  
 }  
 }  
  
 @Override  
 public void onCreate() {  
 super.onCreate();  
 ongakuApp = (OngakuApp) getApplication();  
 housouBinder = new HousouBinder();  
 try {  
 mediaPlayer.reset();  
 hIndex = 0;  
 hPath = ongakuApp.globalOngakus.get(0).getPath();  
 mediaPlayer.setDataSource(hPath);  
 mediaPlayer.prepare();  
 }  
 catch (Exception e) { e.printStackTrace(); }  
 mediaPlayer.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {  
 @Override  
 public void onCompletion(MediaPlayer mediaPlayer) {  
 if (repeat == RipiitoMode.*NO\_LOOP*) {  
 try {  
 mediaPlayer.reset();  
 hPath = ongakuApp.globalOngakus.get(hIndex).getPath();  
 mediaPlayer.setDataSource(hPath);  
 mediaPlayer.prepare();  
 }  
 catch (Exception e) { e.printStackTrace(); }  
 }  
 else if (repeat == RipiitoMode.*SINGLE\_LOOP*) {  
 try {  
 mediaPlayer.reset();  
 hPath = ongakuApp.globalOngakus.get(hIndex).getPath();  
 mediaPlayer.setDataSource(hPath);  
 mediaPlayer.prepare();  
 mediaPlayer.start();  
 }  
 catch (Exception e) { e.printStackTrace(); }  
 }  
 else {  
 if (shuffle == ShaffuruMode.*ORDER*) {  
 housouBinder.next();  
 }  
 else {  
 try {  
 mediaPlayer.reset();  
 hIndex = (int)(Math.*random*() \* ongakuApp.globalOngakus.size());  
 hPath = ongakuApp.globalOngakus.get(hIndex).getPath();  
 mediaPlayer.setDataSource(hPath);  
 mediaPlayer.prepare();  
 mediaPlayer.start();  
 }  
 catch (Exception e) { e.printStackTrace(); }  
  
 }  
 }  
 }  
 });  
 }  
}

OngakuApp.java

package com.leonelacs.ongaku;  
  
import android.app.Application;  
import android.content.ContentResolver;  
import android.database.Cursor;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.net.Uri;  
import android.provider.MediaStore;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class OngakuApp extends Application {  
  
 public List<Ongaku> globalOngakus = new ArrayList<>();  
  
 public void setGlobalOngakus(List<Ongaku> globalOngakus) {  
 this.globalOngakus = globalOngakus;  
 }  
  
 public Ongaku getOngaku(int index) {  
 return globalOngakus.get(index);  
 }  
  
 public int getGlobalOngakusSize() {  
 return globalOngakus.size();  
 }  
  
 @Override  
 public void onCreate() {  
 super.onCreate();  
 globalOngakus = getOngakuFromPhone();  
 }  
  
 public List<Ongaku> getOngakuFromPhone() {  
 ContentResolver contentResolver = getContentResolver();  
 Cursor cursor = contentResolver.query(MediaStore.Audio.Media.*EXTERNAL\_CONTENT\_URI*,null,null,null,MediaStore.Audio.Media.*DEFAULT\_SORT\_ORDER*);  
 List<Ongaku> myuujikkuFiles = new ArrayList<>();  
 if (cursor.moveToFirst()) {  
 do {  
 Ongaku m = new Ongaku();  
 long id = cursor.getLong(cursor.getColumnIndex(MediaStore.Audio.Media.*\_ID*));  
 String title = cursor.getString(cursor.getColumnIndex(MediaStore.Audio.Media.*TITLE*));  
 String artist = cursor.getString(cursor.getColumnIndex(MediaStore.Audio.Media.*ARTIST*));  
 String album = cursor.getString(cursor.getColumnIndex(MediaStore.Audio.Media.*ALBUM*));  
 long albumId = cursor.getLong(cursor.getColumnIndex(MediaStore.Audio.Media.*ALBUM\_ID*));  
 long duration = cursor.getLong(cursor.getColumnIndex(MediaStore.Audio.Media.*DURATION*));  
 long size = cursor.getLong(cursor.getColumnIndex(MediaStore.Audio.Media.*SIZE*));  
 String path = cursor.getString(cursor.getColumnIndex(MediaStore.Audio.Media.*DATA*));  
 int ongakuDesu = cursor.getInt(cursor.getColumnIndex(MediaStore.Audio.Media.*IS\_MUSIC*));  
 Bitmap cover = getKabaa(albumId);  
 if (ongakuDesu != 0 && duration / (500 \* 60) >= 1) {  
 m.setId(id);  
 m.setTitle(title);  
 m.setArtist(artist);  
 m.setAlbum(album);  
 m.setAlbumId(albumId);  
 m.setDuration(duration);  
 m.setSize(size);  
 m.setPath(path);  
 m.setCover(cover);  
 myuujikkuFiles.add(m);  
 }  
 }  
 while (cursor.moveToNext());  
 }  
 cursor.close();  
 return myuujikkuFiles;  
 }  
  
 public Bitmap getKabaa(long albumId) {  
 String mUriAlbums = "content://media/external/audio/albums";  
 String[] projection = new String[]{"album\_art"};  
 Cursor cur = this.getContentResolver().query(Uri.*parse*(mUriAlbums + "/" + Long.*toString*(albumId)), projection, null, null, null);  
 String album\_art = null;  
 if (cur.getCount() > 0 && cur.getColumnCount() > 0) {  
 cur.moveToNext();  
 album\_art = cur.getString(0);  
 }  
 cur.close();  
 Bitmap bm = null;  
 if (album\_art != null) {  
 bm = BitmapFactory.*decodeFile*(album\_art);  
 } else {  
 //bm = BitmapFactory.decodeResource(getResources(), R.drawable.default\_cover\_alpha\_400);  
 bm = null;  
 }  
 return bm;  
 }  
  
  
}

MainActivity.java

package com.leonelacs.ongaku;  
  
import android.content.ComponentName;  
import android.content.ContentResolver;  
import android.content.Context;  
import android.content.Intent;  
import android.content.ServiceConnection;  
import android.database.Cursor;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.net.Uri;  
import android.os.Handler;  
import android.os.IBinder;  
import android.os.Message;  
import android.provider.MediaStore;  
import android.renderscript.Allocation;  
import android.renderscript.Element;  
import android.renderscript.RenderScript;  
import android.renderscript.ScriptIntrinsicBlur;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.support.v7.widget.RecyclerView;  
import android.support.v7.widget.StaggeredGridLayoutManager;  
import android.view.View;  
import android.widget.Button;  
import android.widget.ImageView;  
import android.widget.SeekBar;  
import android.widget.TextView;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class MainActivity extends AppCompatActivity {  
  
 RecyclerView ongakuRecycler;  
 OngakuAdapter ongakuAdapter;  
  
 int cIndex = 0;  
  
 Intent startIntent;  
 HousouService.HousouBinder housouBinder;  
 ServiceConnection serviceConnection;  
  
 ImageView mainAlbumCover;  
 TextView mainTitle;  
 TextView mainArtist;  
 Button mainButtonPrevious;  
 Button mainButtonPlay;  
 Button mainButtonNext;  
 SeekBar mainSeekBar;  
  
 OngakuApp ongakuApp;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 mainAlbumCover = findViewById(R.id.*MainAlbumCover*);  
 mainTitle = findViewById(R.id.*MainTitle*);  
 mainArtist = findViewById(R.id.*MainArtist*);  
 mainButtonPrevious = findViewById(R.id.*MainButtonPrevious*);  
 mainButtonPlay = findViewById(R.id.*MainButtonPlay*);  
 mainButtonNext = findViewById(R.id.*MainButtonNext*);  
 mainSeekBar = findViewById(R.id.*MainSeekBar*);  
  
 ongakuApp = (OngakuApp) getApplication();  
  
 ongakuRecycler = (RecyclerView)findViewById(R.id.*OngakuRecycler*);  
 StaggeredGridLayoutManager staggeredGridLayoutManager = new StaggeredGridLayoutManager(3, StaggeredGridLayoutManager.*VERTICAL*);  
 ongakuRecycler.setLayoutManager(staggeredGridLayoutManager);  
 ongakuAdapter = new OngakuAdapter(ongakuApp.globalOngakus);  
 ongakuRecycler.setAdapter(ongakuAdapter);  
 ongakuAdapter.notifyDataSetChanged();  
  
 mainAlbumCover.setImageResource(R.drawable.*default\_cover\_alpha\_400*);  
 mainTitle.setText("");  
 mainArtist.setText("");  
  
 ongakuAdapter.setOnItemClickListener(new OngakuAdapter.OnItemClickListener() {  
 @Override  
 public void onItemClick(View view, int position, Ongaku ongaku) {  
 cIndex = position;  
 refreshMainThings(position);  
 housouBinder.select(position);  
 }  
 });  
  
 //service  
 serviceConnection = new ServiceConnection() {  
 @Override  
 public void onServiceConnected(ComponentName componentName, IBinder iBinder) {  
 housouBinder = (HousouService.HousouBinder) iBinder;  
 }  
  
 @Override  
 public void onServiceDisconnected(ComponentName componentName) {  
  
 }  
 };  
 startIntent = new Intent(this, HousouService.class);  
 bindService(startIntent, serviceConnection, *BIND\_AUTO\_CREATE*);  
 startService(startIntent);  
  
 updateProgress();  
  
 mainAlbumCover.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Intent intent = new Intent(view.getContext(), DetailActivity.class);  
 intent.putExtra("cPosition", cIndex);  
 intent.putExtra("cProgress", mainSeekBar.getProgress());  
 intent.putExtra("cDuration", housouBinder.getHousouDuration());  
 ShaffuruMode shuffle = housouBinder.getShuffle();  
 RipiitoMode repeat = housouBinder.getRepeat();  
 intent.putExtra("shuffle", shuffle);  
 intent.putExtra("repeat", repeat);  
 view.getContext().startActivity(intent);  
 }  
 });  
  
 mainButtonPlay.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 if (housouBinder.getIsPlaying()) {  
 housouBinder.pause();  
 }  
 else {  
 refreshMainThings(housouBinder.play());  
 }  
 }  
 });  
  
 mainButtonPrevious.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 refreshMainThings(housouBinder.previous());  
 }  
 });  
  
 mainButtonNext.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 refreshMainThings(housouBinder.next());  
 }  
 });  
  
 mainSeekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {  
 @Override  
 public void onProgressChanged(SeekBar seekBar, int i, boolean b) {  
 if (b) {  
 housouBinder.setHousouProgress(i);  
 }  
 }  
  
 @Override  
 public void onStartTrackingTouch(SeekBar seekBar) {  
  
 }  
  
 @Override  
 public void onStopTrackingTouch(SeekBar seekBar) {  
  
 }  
 });  
  
 }  
  
 private Handler handler = new Handler(new Handler.Callback() {  
 @Override  
 public boolean handleMessage(Message msg) {  
 if(cIndex != housouBinder.getHIndex()){  
 cIndex = housouBinder.getHIndex();  
 refreshMainThings(cIndex);  
 }  
 int progress = housouBinder.getHousouProgress();  
 mainSeekBar.setProgress(progress);  
 updateProgress();  
 return true;  
 }  
 });  
  
  
 private void updateProgress() {  
 Message msg = Message.*obtain*();  
 handler.sendMessageDelayed(msg, 1000);  
 }  
  
 void refreshMainThings(int index) {  
 mainTitle.setText(ongakuApp.globalOngakus.get(index).getTitle());  
 mainArtist.setText(ongakuApp.globalOngakus.get(index).getArtist());  
 if (ongakuApp.globalOngakus.get(index).getCover() == null) {  
 mainAlbumCover.setImageResource(R.drawable.*default\_cover\_alpha\_400*);  
 }  
 else {  
 mainAlbumCover.setImageBitmap(ongakuApp.globalOngakus.get(index).getCover());  
 }  
 mainSeekBar.setMax(housouBinder.getHousouDuration());  
 }  
}

DetailActivity.java

package com.leonelacs.ongaku;  
  
import android.content.ComponentName;  
import android.content.Intent;  
import android.content.ServiceConnection;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.graphics.Color;  
import android.graphics.drawable.BitmapDrawable;  
import android.graphics.drawable.ColorDrawable;  
import android.os.Handler;  
import android.os.IBinder;  
import android.os.Message;  
import android.renderscript.Allocation;  
import android.renderscript.Element;  
import android.renderscript.RenderScript;  
import android.renderscript.ScriptIntrinsicBlur;  
import android.support.constraint.ConstraintLayout;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.text.Layout;  
import android.view.View;  
import android.view.WindowManager;  
import android.widget.Button;  
import android.widget.ImageView;  
import android.widget.SeekBar;  
import android.widget.TextView;  
  
public class DetailActivity extends AppCompatActivity {  
  
 ImageView detailAlbumCover;  
 TextView detailTitle;  
 TextView detailArtist;  
 TextView detailAlbumName;  
 TextView detailCurrent;  
 TextView detailFull;  
 SeekBar detailSeekBar;  
  
 Button detailButtonShaffuru;  
 Button detailButtonPrevious;  
 Button detailButtonPlay;  
 Button detailButtonNext;  
 Button detailButtonRipiito;  
  
 OngakuApp ongakuApp;  
  
 Intent startIntent;  
 HousouService.HousouBinder housouBinder;  
 ServiceConnection serviceConnection;  
  
 int cIndex = 0;  
 ShaffuruMode shuffle = ShaffuruMode.*ORDER*;  
 RipiitoMode repeat = RipiitoMode.*ALL\_LOOP*;  
  
 Bitmap back;  
  
  
  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_detail*);  
  
 ongakuApp = (OngakuApp) getApplication();  
  
 detailAlbumCover = findViewById(R.id.*DetailAlbumCover*);  
 detailTitle = findViewById(R.id.*DetailTitle*);  
 detailArtist = findViewById(R.id.*DetailArtist*);  
 detailAlbumName = findViewById(R.id.*DetailAlbumName*);  
 detailCurrent = findViewById(R.id.*DetailCurrent*);  
 detailFull = findViewById(R.id.*DetailFull*);  
 detailSeekBar = findViewById(R.id.*DetailSeekBar*);  
  
 detailButtonShaffuru = findViewById(R.id.*DetailButtonShaffuru*);  
 detailButtonPrevious = findViewById(R.id.*DetailButtonPrevious*);  
 detailButtonPlay = findViewById(R.id.*DetailButtonPlay*);  
 detailButtonNext = findViewById(R.id.*DetailButtonNext*);  
 detailButtonRipiito = findViewById(R.id.*DetailButtonRipiito*);  
  
 //service  
 serviceConnection = new ServiceConnection() {  
 @Override  
 public void onServiceConnected(ComponentName componentName, IBinder iBinder) {  
 housouBinder = (HousouService.HousouBinder) iBinder;  
 }  
  
 @Override  
 public void onServiceDisconnected(ComponentName componentName) {  
  
 }  
 };  
 startIntent = new Intent(this, HousouService.class);  
 bindService(startIntent, serviceConnection, *BIND\_AUTO\_CREATE*);  
 startService(startIntent);  
  
 int cPosition = getIntent().getIntExtra("cPosition", 0);  
 int cProgress = getIntent().getIntExtra("cProgress", 0);  
 int cDuration = getIntent().getIntExtra("cDuration", 0);  
 shuffle = (ShaffuruMode) getIntent().getSerializableExtra("shuffle");  
 repeat = (RipiitoMode) getIntent().getSerializableExtra("repeat");  
  
 cIndex = cPosition;  
  
 if (ongakuApp.globalOngakus.get(cPosition).getCover() != null) {  
 detailAlbumCover.setImageBitmap(ongakuApp.globalOngakus.get(cPosition).getCover());  
 back = Bitmap.*createBitmap*(ongakuApp.globalOngakus.get(cPosition).getCover());  
 back = blur3Times(back);  
 findViewById(R.id.*DetailBackground*).setBackground(new BitmapDrawable(getResources(), back));  
 }  
 else {  
 detailAlbumCover.setImageResource(R.drawable.*default\_cover\_alpha*);  
 back = BitmapFactory.*decodeResource*(getResources(),R.drawable.*default\_cover\_alpha\_400*);  
 findViewById(R.id.*DetailBackground*).setBackground(getDrawable(R.drawable.*default\_cover*));  
 }  
  
 detailTitle.setText(ongakuApp.globalOngakus.get(cPosition).getTitle());  
 detailArtist.setText(ongakuApp.globalOngakus.get(cPosition).getArtist());  
 detailAlbumName.setText(ongakuApp.globalOngakus.get(cPosition).getAlbum());  
 detailCurrent.setText(toTimeFormat(cProgress));  
 detailFull.setText(toTimeFormat(cDuration));  
 detailSeekBar.setProgress(cProgress);  
 detailSeekBar.setMax(cDuration);  
  
 updateProgress();  
  
 detailButtonPlay.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 if (housouBinder.getIsPlaying()) {  
 housouBinder.pause();  
 }  
 else {  
 refreshDetailThings(housouBinder.play());  
 }  
 }  
 });  
  
 detailButtonPrevious.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 refreshDetailThings(housouBinder.previous());  
 }  
 });  
  
 detailButtonNext.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 refreshDetailThings(housouBinder.next());  
 }  
 });  
  
 detailButtonShaffuru.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 shuffle = housouBinder.changeShuffle();  
 if (shuffle == ShaffuruMode.*ORDER*) {  
 detailButtonShaffuru.setText("SEQ");  
 }  
 else {  
 detailButtonShaffuru.setText("RAN");  
 }  
 }  
 });  
  
 detailButtonRipiito.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 repeat = housouBinder.changeRepeat();  
 if (repeat == RipiitoMode.*NO\_LOOP*) {  
 detailButtonRipiito.setText("OFF");  
 }  
 else if (repeat == RipiitoMode.*SINGLE\_LOOP*) {  
 detailButtonRipiito.setText("ONE");  
 }  
 else {  
 detailButtonRipiito.setText("ALL");  
 }  
 }  
 });  
  
 detailSeekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {  
 @Override  
 public void onProgressChanged(SeekBar seekBar, int i, boolean b) {  
 if (b) {  
 housouBinder.setHousouProgress(i);  
 }  
 }  
  
 @Override  
 public void onStartTrackingTouch(SeekBar seekBar) {  
  
 }  
  
 @Override  
 public void onStopTrackingTouch(SeekBar seekBar) {  
  
 }  
 });  
  
 }  
  
 void refreshDetailThings(int index) {  
 detailTitle.setText(ongakuApp.globalOngakus.get(index).getTitle());  
 detailArtist.setText(ongakuApp.globalOngakus.get(index).getArtist());  
 detailAlbumName.setText(ongakuApp.globalOngakus.get(index).getAlbum());  
 if (ongakuApp.globalOngakus.get(index).getCover() == null) {  
 detailAlbumCover.setImageResource(R.drawable.*default\_cover\_alpha*);  
 findViewById(R.id.*DetailBackground*).setBackground(getDrawable(R.drawable.*default\_cover*));  
 }  
 else {  
 detailAlbumCover.setImageBitmap(ongakuApp.globalOngakus.get(index).getCover());  
 back = ongakuApp.globalOngakus.get(index).getCover();  
 back = blur3Times(back);  
 findViewById(R.id.*DetailBackground*).setBackground(new BitmapDrawable(getResources(), back));  
 }  
 detailSeekBar.setMax(housouBinder.getHousouDuration());  
 detailFull.setText(toTimeFormat(housouBinder.getHousouDuration()));  
  
 }  
  
 private Handler handler = new Handler(new Handler.Callback() {  
 @Override  
 public boolean handleMessage(Message msg) {  
 if(cIndex != housouBinder.getHIndex()){  
 cIndex = housouBinder.getHIndex();  
 refreshDetailThings(cIndex);  
 }  
 int progress = housouBinder.getHousouProgress();  
 detailSeekBar.setProgress(progress);  
 detailCurrent.setText(toTimeFormat(housouBinder.getHousouProgress()));  
 updateProgress();  
 return true;  
 }  
 });  
  
  
 private void updateProgress() {  
 Message msg = Message.*obtain*();  
 handler.sendMessageDelayed(msg, 1000);  
 }  
  
 String toTimeFormat(int time\_mm) {  
 time\_mm /= 1000;  
 int full\_min = time\_mm / 60;  
 int full\_sec = time\_mm % 60;  
 String str\_full\_min = "", str\_full\_sec = "";  
 if (full\_min < 10) {  
 str\_full\_min = "0";  
 }  
 if (full\_sec < 10) {  
 str\_full\_sec = "0";  
 }  
 str\_full\_min = str\_full\_min + String.*valueOf*(full\_min);  
 str\_full\_sec = str\_full\_sec + String.*valueOf*(full\_sec);  
 return str\_full\_min + ":" + str\_full\_sec;  
 }  
 String toTimeFormat(long time\_mm) {  
 time\_mm /= 1000;  
 long full\_min = time\_mm / 60;  
 long full\_sec = time\_mm % 60;  
 String str\_full\_min = "", str\_full\_sec = "";  
 if (full\_min < 10) {  
 str\_full\_min = "0";  
 }  
 if (full\_sec < 10) {  
 str\_full\_sec = "0";  
 }  
 str\_full\_min = str\_full\_min + String.*valueOf*(full\_min);  
 str\_full\_sec = str\_full\_sec + String.*valueOf*(full\_sec);  
 return str\_full\_min + ":" + str\_full\_sec;  
 }  
  
 Bitmap blurBitmap(Bitmap bmp) {  
 Bitmap bitmap = Bitmap.*createBitmap*(bmp);  
 Bitmap blurred = Bitmap.*createBitmap*(bitmap.getWidth(), bitmap.getHeight(), Bitmap.Config.*ARGB\_8888*);  
 RenderScript rs = RenderScript.*create*(this);  
 ScriptIntrinsicBlur blur = ScriptIntrinsicBlur.*create*(rs, Element.*U8\_4*(rs));  
 Allocation in = Allocation.*createFromBitmap*(rs, bitmap);  
 Allocation out = Allocation.*createFromBitmap*(rs, blurred);  
 blur.setRadius(25f);  
 blur.setInput(in);  
 blur.forEach(out);  
 out.copyTo(blurred);  
 rs.destroy();  
 return blurred;  
 }  
  
 Bitmap blur3Times(Bitmap bmp) {  
 Bitmap blurred = blurBitmap(bmp);  
 blurred = blurBitmap(blurred);  
 blurred = blurBitmap(blurred);  
 return blurred;  
 }  
}