作业四

实验截图：



说明：

通过读取手机内部sd卡文件播放音频，同时也可以通过网络url播放音频

关键代码：

musicList.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

index = position;

play(position);

}

});

imageButtonPre.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

int idx = index -1 >=0?index-1:0;

index = idx;

play(index);

}

});

imageButtonNext.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

int idx = index + 1 >= musicLists.length ? musicLists.length:index+1;

index = idx;

play(index);

}

});

musicList.setAdapter(arrayAdapter);

if (mediaPlayer == null) return;

mediaPlayer.setVolume(0.5f, 0.5f);

if (mediaPlayer.isPlaying()) {

buttonPlayOrPause.setImageResource(android.R.drawable.ic\_media\_pause);

} else if (!mediaPlayer.isPlaying()) {

buttonPlayOrPause.setImageResource(android.R.drawable.ic\_media\_play);

}

buttonForward.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (mediaPlayer != null && mediaPlayer.isPlaying()) {

mediaPlayer.seekTo((mediaPlayer.getCurrentPosition() + 10000) > mediaPlayer.getDuration() ? mediaPlayer.getCurrentPosition() : (mediaPlayer.getCurrentPosition() + 10000));

}

}

});

buttonBack.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (mediaPlayer != null && mediaPlayer.isPlaying()) {

mediaPlayer.seekTo((mediaPlayer.getCurrentPosition() - 10000) < 0 ? 0 : (mediaPlayer.getCurrentPosition() - 10000));

}

}

});

seekBarVolume.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {

@Override

public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {

if (mediaPlayer != null) {

mediaPlayer.setVolume(seekBarVolume.getProgress() / 100.0f, seekBarVolume.getProgress() / 100.0f);

}

}

@Override

public void onStartTrackingTouch(SeekBar seekBar) {

}

@Override

public void onStopTrackingTouch(SeekBar seekBar) {

}

});

buttonPlayOrPause.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (mediaPlayer != null && mediaPlayer.isPlaying()) {

mediaPlayer.pause();

buttonPlayOrPause.setImageResource(android.R.drawable.ic\_media\_play);

} else if (mediaPlayer != null && mediaPlayer.isPlaying() == false) {

mediaPlayer.start();

buttonPlayOrPause.setImageResource(android.R.drawable.ic\_media\_pause);

}

}

});

seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {

@Override

public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {

if (mediaPlayer != null && mediaPlayer.isPlaying() && fromUser) {

mediaPlayer.seekTo((int) (mediaPlayer.getDuration() \* progress / 100));

}

}

@Override

public void onStartTrackingTouch(SeekBar seekBar) {

}

@Override

public void onStopTrackingTouch(SeekBar seekBar) {

}

});

new MyTask().execute();

}

@Override

public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {

super.onRequestPermissionsResult(requestCode, permissions, grantResults);

if (requestCode == 1) {

play(0);

}

}

class MyTask extends AsyncTask<Object, Integer, Object> {

@Override

protected Object doInBackground(Object... objects) {

while (flag) {

try {

Thread.sleep(100);

} catch (InterruptedException e) {

e.printStackTrace();

}

publishProgress(1);

}

return null;

}

@Override

protected void onPreExecute() {

super.onPreExecute();

}

@Override

protected void onPostExecute(Object o) {

super.onPostExecute(o);

}

@Override

protected void onProgressUpdate(Integer... values) {

super.onProgressUpdate(values);

if (mediaPlayer == null || mediaPlayer.isPlaying() == false) return;

int duration = mediaPlayer.getDuration();

Log.i("DUR", duration + "");

int current = mediaPlayer.getCurrentPosition();

Log.i("CURR", current + "");

seekBar.setProgress(Math.round((current + 0.0f) / duration \* 100));

Log.i("PRO", Math.round(current / duration \* 100) + "");

String str =musicLists[index]+"\n";

str += " 总时长：" + (duration / 1000 / 60) + "分" + (duration / 1000 % 60) + "秒"+"\n";

str += " 当前进度：" + (current / 1000 / 60) + "分" + (current / 1000 % 60) + "秒"+"\n";

textView.setText(str);

float degree = cdImage.getRotation()+20;

if(degree>= 360){

degree = 0;

}

cdImage.setRotation(degree);

if(duration/1000 == current/1000){

if(index!=6) {

play(index + 1);

index++;

}else{

play(0);

index = 0;

}

}

}

}

private int getRawResourceId(String resourceName) {

return this.getResources().getIdentifier(resourceName, "raw", this.getPackageName());

}

private void play(int index) {

// Initialize the MediaPlayer with the music file from the raw directory

if (mediaPlayer!=null) {

if(mediaPlayer.isPlaying()){

mediaPlayer.stop();

}

mediaPlayer.reset();

mediaPlayer.release();

mediaPlayer = null;

}

mediaPlayer = new MediaPlayer();

mediaPlayer = MediaPlayer.create(this, getRawResourceId(musicLists[index]));

mediaPlayer.setAudioStreamType(AudioManager.STREAM\_MUSIC);

mediaPlayer.start();

}

private void play(String path) throws IOException {

// Initialize the MediaPlayer with the music file from the raw directory

if (mediaPlayer!=null) {

if(mediaPlayer.isPlaying()){

mediaPlayer.stop();

}

mediaPlayer.reset();

mediaPlayer.release();

mediaPlayer = null;

}

mediaPlayer = new MediaPlayer();

mediaPlayer.setDataSource(path);

mediaPlayer.start();

}

@Override

protected void onDestroy() {

super.onDestroy();

if (mediaPlayer != null) {

mediaPlayer.release();

mediaPlayer = null;

}

}

}