[**Android多媒体学习十二：Android中Video的三种播放方式的实现**](http://blog.csdn.net/chenjie19891104/article/details/6338202)

分类： [Android 学习笔记](http://blog.csdn.net/chenjie19891104/article/category/756236) 2011-04-21 11:29 1240人阅读 [评论](http://blog.csdn.net/chenjie19891104/article/details/6338202#comments)(2) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/chenjie19891104/article/details/6338202#report)

在Android中，我们有三种方式来实现视频的播放。

1、使用其自带的播放器。指定Action为ACTION\_VIEW,Data为Uri，Type为其MIME类型。

2、使用VideoView来播放。在布局文件中使用VideoView结合MediaController来实现对其控制。

3、使用MediaPlayer类和SurfaceView来实现。这种方式最灵活，也最复杂。

下面给出这三种方式，

1、调用其自带的播放器：

[view plaincopy to clipboardprint?](http://blog.csdn.net/chenjie19891104/article/details/6338202)

1. Uri uri = Uri.parse(Environment.getExternalStorageDirectory().getPath()+"/Test\_Movie.m4v");
2. //调用系统自带的播放器
3. Intent intent = new Intent(Intent.ACTION\_VIEW);
4. Log.v("URI:::::::::", uri.toString());
5. intent.setDataAndType(uri, "video/mp4");
6. startActivity(intent);



2、使用VideoView来实现：

[view plaincopy to clipboardprint?](http://blog.csdn.net/chenjie19891104/article/details/6338202)

1. Uri uri = Uri.parse(Environment.getExternalStorageDirectory().getPath()+"/Test\_Movie.m4v");
2. VideoView videoView = (VideoView)this.findViewById(R.id.video\_view);
3. videoView.setMediaController(new MediaController(this));
4. videoView.setVideoURI(uri);
5. //videoView.start();
6. videoView.requestFocus();



3、使用MediaPlayer：

[view plaincopy to clipboardprint?](http://blog.csdn.net/chenjie19891104/article/details/6338202)

1. package demo.camera;
2. import java.io.IOException;
3. import android.app.Activity;
4. import android.media.MediaPlayer;
5. import android.media.MediaPlayer.OnCompletionListener;
6. import android.media.MediaPlayer.OnErrorListener;
7. import android.media.MediaPlayer.OnInfoListener;
8. import android.media.MediaPlayer.OnPreparedListener;
9. import android.media.MediaPlayer.OnSeekCompleteListener;
10. import android.media.MediaPlayer.OnVideoSizeChangedListener;
11. import android.os.Bundle;
12. import android.os.Environment;
13. import android.util.Log;
14. import android.view.Display;
15. import android.view.SurfaceHolder;
16. import android.view.SurfaceView;
17. import android.widget.LinearLayout;
18. /\*\*
19. \* 该实例中使用MediaPlayer完成播放，同时界面使用SurfaceView来实现
20. \*
21. \* 这里我们实现MediaPlayer中很多状态变化时的监听器
22. \*
23. \* 使用Mediaplayer时，也可以使用MediaController类，但是需要实现MediaController.mediaController接口
24. \* 实现一些控制方法。
25. \*
26. \* 然后，设置controller.setMediaPlayer(),setAnchorView(),setEnabled(),show()就可以了，这里不再实现
27. \* @author Administrator
28. \*
29. \*/
30. public class VideoSurfaceDemo extends Activity implements OnCompletionListener,OnErrorListener,OnInfoListener,
31. OnPreparedListener, OnSeekCompleteListener,OnVideoSizeChangedListener,SurfaceHolder.Callback{
32. private Display currDisplay;
33. private SurfaceView surfaceView;
34. private SurfaceHolder holder;
35. private MediaPlayer player;
36. private int vWidth,vHeight;
37. //private boolean readyToPlay = false;

40. public void onCreate(Bundle savedInstanceState){
41. super.onCreate(savedInstanceState);
42. this.setContentView(R.layout.video\_surface);

45. surfaceView = (SurfaceView)this.findViewById(R.id.video\_surface);
46. //给SurfaceView添加CallBack监听
47. holder = surfaceView.getHolder();
48. holder.addCallback(this);
49. //为了可以播放视频或者使用Camera预览，我们需要指定其Buffer类型
50. holder.setType(SurfaceHolder.SURFACE\_TYPE\_PUSH\_BUFFERS);
52. //下面开始实例化MediaPlayer对象
53. player = new MediaPlayer();
54. player.setOnCompletionListener(this);
55. player.setOnErrorListener(this);
56. player.setOnInfoListener(this);
57. player.setOnPreparedListener(this);
58. player.setOnSeekCompleteListener(this);
59. player.setOnVideoSizeChangedListener(this);
60. Log.v("Begin:::", "surfaceDestroyed called");
61. //然后指定需要播放文件的路径，初始化MediaPlayer
62. String dataPath = Environment.getExternalStorageDirectory().getPath()+"/Test\_Movie.m4v";
63. try {
64. player.setDataSource(dataPath);
65. Log.v("Next:::", "surfaceDestroyed called");
66. } catch (IllegalArgumentException e) {
67. // TODO Auto-generated catch block
68. e.printStackTrace();
69. } catch (IllegalStateException e) {
70. // TODO Auto-generated catch block
71. e.printStackTrace();
72. } catch (IOException e) {
73. // TODO Auto-generated catch block
74. e.printStackTrace();
75. }
77. //然后，我们取得当前Display对象
78. currDisplay = this.getWindowManager().getDefaultDisplay();
79. }
81. @Override
82. public void surfaceChanged(SurfaceHolder arg0, int arg1, int arg2, int arg3) {
83. // 当Surface尺寸等参数改变时触发
84. Log.v("Surface Change:::", "surfaceChanged called");
85. }
86. @Override
87. public void surfaceCreated(SurfaceHolder holder) {
88. // 当SurfaceView中的Surface被创建的时候被调用
89. //在这里我们指定MediaPlayer在当前的Surface中进行播放
90. player.setDisplay(holder);
91. //在指定了MediaPlayer播放的容器后，我们就可以使用prepare或者prepareAsync来准备播放了
92. player.prepareAsync();
94. }
95. @Override
96. public void surfaceDestroyed(SurfaceHolder holder) {
98. Log.v("Surface Destory:::", "surfaceDestroyed called");
99. }
100. @Override
101. public void onVideoSizeChanged(MediaPlayer arg0, int arg1, int arg2) {
102. // 当video大小改变时触发
103. //这个方法在设置player的source后至少触发一次
104. Log.v("Video Size Change", "onVideoSizeChanged called");
106. }
107. @Override
108. public void onSeekComplete(MediaPlayer arg0) {
109. // seek操作完成时触发
110. Log.v("Seek Completion", "onSeekComplete called");
112. }
113. @Override
114. public void onPrepared(MediaPlayer player) {
115. // 当prepare完成后，该方法触发，在这里我们播放视频
117. //首先取得video的宽和高
118. vWidth = player.getVideoWidth();
119. vHeight = player.getVideoHeight();
121. if(vWidth > currDisplay.getWidth() || vHeight > currDisplay.getHeight()){
122. //如果video的宽或者高超出了当前屏幕的大小，则要进行缩放
123. float wRatio = (float)vWidth/(float)currDisplay.getWidth();
124. float hRatio = (float)vHeight/(float)currDisplay.getHeight();
126. //选择大的一个进行缩放
127. float ratio = Math.max(wRatio, hRatio);
129. vWidth = (int)Math.ceil((float)vWidth/ratio);
130. vHeight = (int)Math.ceil((float)vHeight/ratio);
132. //设置surfaceView的布局参数
133. surfaceView.setLayoutParams(new LinearLayout.LayoutParams(vWidth, vHeight));
135. //然后开始播放视频
137. player.start();
138. }
139. }
140. @Override
141. public boolean onInfo(MediaPlayer player, int whatInfo, int extra) {
142. // 当一些特定信息出现或者警告时触发
143. switch(whatInfo){
144. case MediaPlayer.MEDIA\_INFO\_BAD\_INTERLEAVING:
145. break;
146. case MediaPlayer.MEDIA\_INFO\_METADATA\_UPDATE:
148. break;
149. case MediaPlayer.MEDIA\_INFO\_VIDEO\_TRACK\_LAGGING:
151. break;
152. case MediaPlayer.MEDIA\_INFO\_NOT\_SEEKABLE:
154. break;
155. }
156. return false;
157. }
158. @Override
159. public boolean onError(MediaPlayer player, int whatError, int extra) {
160. Log.v("Play Error:::", "onError called");
161. switch (whatError) {
162. case MediaPlayer.MEDIA\_ERROR\_SERVER\_DIED:
163. Log.v("Play Error:::", "MEDIA\_ERROR\_SERVER\_DIED");
164. break;
165. case MediaPlayer.MEDIA\_ERROR\_UNKNOWN:
166. Log.v("Play Error:::", "MEDIA\_ERROR\_UNKNOWN");
167. break;
168. default:
169. break;
170. }
171. return false;
172. }
173. @Override
174. public void onCompletion(MediaPlayer player) {
175. // 当MediaPlayer播放完成后触发
176. Log.v("Play Over:::", "onComletion called");
177. this.finish();
178. }
179. }

public class VideoActivity extends Activity {

private static final String TAG = "VideoActivity";

private EditText filenameText;

private SurfaceView surfaceView;

private MediaPlayer mediaPlayer;

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

ButtonClickListener listener = new ButtonClickListener();

ImageButton playButton = (ImageButton) this.findViewById(R.id.play);

playButton.setOnClickListener(listener);

ImageButton pauseButton = (ImageButton) this.findViewById(R.id.pause);

pauseButton.setOnClickListener(listener);

ImageButton resetButton = (ImageButton) this.findViewById(R.id.reset);

resetButton.setOnClickListener(listener);

ImageButton stopButton = (ImageButton) this.findViewById(R.id.stop);

stopButton.setOnClickListener(listener);

// 重点 这个地方需要设置回调函数 继承Callback 类实现三个方法

surfaceView.getHolder().addCallback(new myCallback());

filenameText = (EditText) this.findViewById(R.id.filename);

surfaceView = (SurfaceView) findViewById(R.id.surfaceView);

surfaceView.getHolder() .setType(SurfaceHolder.SURFACE\_TYPE\_PUSH\_BUFFERS);

// 设置这个大小

surfaceView.getHolder().setFixedSize(176, 144);

mediaPlayer = new MediaPlayer();

}

private class myCallback implements android.view.SurfaceHolder.Callback {

@Override

public void surfaceChanged(SurfaceHolder holder, int format, int width,

int height) {

// TODO Auto-generated method stub

}

@Override

public void surfaceCreated(SurfaceHolder holder) {

// TODO Auto-generated method stub

}

@Override

public void surfaceDestroyed(SurfaceHolder holder) {

// TODO Auto-generated method stub

}

}

// 当来电话的时候需要停掉播放业务

@Override

protected void onPause() {

if (mediaPlayer.isPlaying())

mediaPlayer.pause();

super.onPause();

}

@Override

protected void onResume() {

if (mediaPlayer.isPlaying())

mediaPlayer.pause();

super.onResume();

}

@Override

protected void onDestroy() {

if (mediaPlayer.isPlaying())

mediaPlayer.stop();

// 注意这个地方是释放

mediaPlayer.release();

super.onStop();

}

private class ButtonClickListener implements View.OnClickListener {

@Override

public void onClick(View v) {

try {

String filename = filenameText.getText().toString();

switch (v.getId()) {

case R.id.play:

// 得到文件名称 重点代码

String filenames = filenameText.getText().toString();

// 第一步对给对象进行重设

mediaPlayer.reset();

// 设置流的类型 关键代码

mediaPlayer.setAudioStreamType(AudioManager.STREAM\_MUSIC);

// 设置数据来源

mediaPlayer.setDataSource("/sdcard/" + filenames); // 设置需要播放的视频

// 把视频输出到显示控件上 重点代码

mediaPlayer.setDisplay(surfaceView.getHolder());

// 播放时前需要缓冲一下

mediaPlayer.prepare();

mediaPlayer.start();

break;

case R.id.pause:

if (mediaPlayer.isPlaying()) {

mediaPlayer.pause();

} else {

mediaPlayer.start();

}

break;

case R.id.reset:

if (!mediaPlayer.isPlaying()) {

mediaPlayer.start();

}

// 跳回到起始的地方重新播放

mediaPlayer.seekTo(0);

break;

case R.id.stop:

if (mediaPlayer.isPlaying()) {

mediaPlayer.stop();

}

break;

}

} catch (Exception e) {

Log.e(TAG, e.toString());

}

}

}

}