

## 原 最简单的基于FFmpeg的移动端例子：Android 视频解码器

2015年07月24日 19:02:29 阅读数：39227

=====

最简单的基于FFmpeg的移动端例子系列文章列表：

[最简单的基于FFmpeg的移动端例子：Android HelloWorld](#)

[最简单的基于FFmpeg的移动端例子：Android 视频解码器](#)

[最简单的基于FFmpeg的移动端例子：Android 视频解码器-单个库版](#)

[最简单的基于FFmpeg的移动端例子：Android 推流器](#)

[最简单的基于FFmpeg的移动端例子：Android 视频转码器](#)

[最简单的基于FFmpeg的移动端例子附件：Android 自带播放器](#)

[最简单的基于FFmpeg的移动端例子附件：SDL Android HelloWorld](#)

[最简单的基于FFmpeg的移动端例子：IOS HelloWorld](#)

[最简单的基于FFmpeg的移动端例子：IOS 视频解码器](#)

[最简单的基于FFmpeg的移动端例子：IOS 推流器](#)

[最简单的基于FFmpeg的移动端例子：IOS 视频转码器](#)

[最简单的基于FFmpeg的移动端例子附件：IOS自带播放器](#)

[最简单的基于FFmpeg的移动端例子：Windows Phone HelloWorld](#)

=====

本文记录一个安卓平台下基于FFmpeg的视频解码器。该视频解码器C语言的源代码来自于《 [最简单的基于FFMPEG+SDL的视频播放器](#) 》。相关的概念就不再重复记录了。

□

## 源代码

项目的目录结构如图所示。Java源代码位于src目录，而C代码位于jni目录。

□

Android程序Java端代码位于src\com\leixiaohua1020\sffmpegandroiddecoder\MainActivity.java，如下所示。

```
1.  /**
2.   * 最简单的基于FFmpeg的视频解码器-安卓
3.   * Simplest FFmpeg Android Decoder
4.   *
5.   * 雷霄骅 Lei Xiaohua
6.   * leixiaohua1020@126.com
7.   * 中国传媒大学/数字电视技术
8.   * Communication University of China / Digital TV Technology
9.   * http://blog.csdn.net/leixiaohua1020
10.  *
11.  * 本程序是安卓平台下最简单的基于FFmpeg的视频解码器。它可以将输入的视频数据解码成YUV像素数据。
12.  *
13.  * This software is the simplest decoder based on FFmpeg in Android. It can decode video stream
14.  * to raw YUV data.
15.  *
16.  */
17. package com.leixiaohua1020.sffmpegandroiddecoder;
18.
19.
20. import android.os.Bundle;
21. import android.os.Environment;
22. import android.app.Activity;
23. import android.text.Editable;
24. import android.util.Log;
25. import android.view.Menu;
26. import android.view.View;
27. import android.view.View.OnClickListener;
28. import android.widget.Button;
29. import android.widget.EditText;
30. import android.widget.TextView;
31.
32. public class MainActivity extends Activity {
33.
34.
35.
36.     @Override
37.     protected void onCreate(Bundle savedInstanceState) {
38.         super.onCreate(savedInstanceState);
39.         setContentView(R.layout.activity_main);
40.
41.         Button startButton = (Button) this.findViewById(R.id.button_start);
42.         final EditText urlEditText_input= (EditText) this.findViewById(R.id.input_url);
43.         final EditText urlEditText_output= (EditText) this.findViewById(R.id.output_url);
44.
45.         startButton.setOnClickListener(new OnClickListener() {
46.             public void onClick(View arg0){
47.
48.                 String folderurl=Environment.getExternalStorageDirectory().getPath();
49.
50.                 String urltext_input=urlEditText_input.getText().toString();
51.                 String inputurl=folderurl+"/"+urltext_input;
52.
53.                 String urltext_output=urlEditText_output.getText().toString();
54.                 String outputurl=folderurl+"/"+urltext_output;
55.
56.                 Log.i("inputurl",inputurl);
57.                 Log.i("outputurl",outputurl);
58.
59.                 decode(inputurl,outputurl);
60.
61.             }
62.         });
63.     }
64.
65.     @Override
66.     public boolean onCreateOptionsMenu(Menu menu) {
67.         // Inflate the menu; this adds items to the action bar if it is present.
68.         getMenuInflater().inflate(R.menu.main, menu);
69.         return true;
70.     }
71.
72.     //JNI
73.     public native int decode(String inputurl, String outputurl);
74.
75.     static{
76.         System.loadLibrary("avutil-54");
77.         System.loadLibrary("swresample-1");
78.         System.loadLibrary("avcodec-56");
79.         System.loadLibrary("avformat-56");
80.         System.loadLibrary("swscale-3");
81.         System.loadLibrary("postproc-53");
82.         System.loadLibrary("avfilter-5");
83.         System.loadLibrary("avdevice-56");
84.         System.loadLibrary("sffdecoder");
85.     }
86. }
```

C语言端源代码位于jni/simplest\_ffmpeg\_decoder.c，如下所示。

```
[cpp]  
1.  /**
2.   * 最简单的基于FFmpeg的视频解码器-安卓
3.   * Simplest FFMpeg Android Decoder
4.   *
5.   * 雷霄骅 Lei Xiaohua
6.   * leixiaohua1020@126.com
7.   * 中国传媒大学/数字电视技术
8.   * Communication University of China / Digital TV Technology
9.   * http://blog.csdn.net/leixiaohua1020
10.  *
11.  * 本程序是安卓平台下最简单的基于FFmpeg的视频解码器。它可以将输入的视频数据解码成YUV像素数据。
12.  *
13.  * This software is the simplest decoder based on FFmpeg in Android. It can decode video stream
14.  * to raw YUV data.
15.  *
16.  */
17.
18.
19. #include <stdio.h>
20. #include <time.h>
21.
22. #include "libavcodec/avcodec.h"
23. #include "libavformat/avformat.h"
24. #include "libswscale/swscale.h"
25. #include "libavutil/log.h"
26.
27. #ifdef ANDROID
28. #include <jni.h>
29. #include <android/log.h>
30. #define LOGE(format, ...) __android_log_print(ANDROID_LOG_ERROR, ">_<", format, ## __VA_ARGS__)
31. #define LOGI(format, ...) __android_log_print(ANDROID_LOG_INFO, "^_^", format, ## __VA_ARGS__)
32. #else
33. #define LOGE(format, ...) printf(">_< " format "\n", ## __VA_ARGS__)
34. #define LOGI(format, ...) printf("^_^ " format "\n", ## __VA_ARGS__)
35. #endif
36.
37.
38. //Output FFmpeg's av_log()
39. void custom_log(void *ptr, int level, const char* fmt, va_list vl){
40.     FILE *fp=fopen("/storage/emulated/0/av_log.txt","a+");
41.     if(fp){
42.         vfprintf(fp,fmt,vl);
43.         fflush(fp);
44.         fclose(fp);
45.     }
46. }
47.
48. JNIEXPORT jint JNICALL Java_com_leixiaohua1020_sffmpegandroiddecoder_MainActivity_decode
49. (JNIEnv *env, jobject obj, jstring input_jstr, jstring output_jstr)
50. {
51.     AVFormatContext *pFormatCtx;
52.     int i, videoindex;
53.     AVCodecContext *pCodecCtx;
54.     AVCodec *pCodec;
55.     AVFrame *pFrame,*pFrameYUV;
56.     uint8_t *out_buffer;
57.     AVPacket *packet;
58.     int y_size;
59.     int ret, got_picture;
60.     struct SwsContext *img_convert_ctx;
61.     FILE *fp_yuv;
62.     int frame_cnt;
63.     clock_t time_start, time_finish;
64.     double time_duration = 0.0;
65.
66.     char input_str[500]={0};
67.     char output_str[500]={0};
68.     char info[1000]={0};
69.     sprintf(input_str,"%s",(*env)->GetStringUTFChars(env,input_jstr, NULL));
70.     sprintf(output_str,"%s",(*env)->GetStringUTFChars(env,output_jstr, NULL));
71.
72.     //FFmpeg av_log() callback
73.     av_log_set_callback(custom_log);
74.
75.     av_register_all();
76.     avformat_network_init();
77.     pFormatCtx = avformat_alloc_context();
78.
79.     if(avformat_open_input(&pFormatCtx,input_str,NULL,NULL)!=0){
80.         LOGE("Couldn't open input stream.\n");
81.         return -1;
82.     }
83.     if(avformat_find_stream_info(pFormatCtx,NULL)<0){
84.         LOGE("Couldn't find stream information.\n");
85.         return -1;
86.     }
87.     videoindex=-1;
```

```

88.     for(i=0; i<pFormatCtx->nb_streams; i++)
89.         if(pFormatCtx->streams[i]->codec->codec_type==AVMEDIA_TYPE_VIDEO){
90.             videoindex=i;
91.             break;
92.         }
93.     if(videoindex==-1){
94.         LOGE("Couldn't find a video stream.\n");
95.         return -1;
96.     }
97.     pCodecCtx=pFormatCtx->streams[videoindex]->codec;
98.     pCodec=avcodec_find_decoder(pCodecCtx->codec_id);
99.     if(pCodec==NULL){
100.         LOGE("Couldn't find Codec.\n");
101.         return -1;
102.     }
103.     if(avcodec_open2(pCodecCtx, pCodec,NULL)<0){
104.         LOGE("Couldn't open codec.\n");
105.         return -1;
106.     }
107.
108.     pFrame=av_frame_alloc();
109.     pFrameYUV=av_frame_alloc();
110.     out_buffer=(unsigned char *)av_malloc(av_image_get_buffer_size(AV_PIX_FMT_YUV420P, pCodecCtx->width, pCodecCtx->height,1));
111.     av_image_fill_arrays(pFrameYUV->data, pFrameYUV->linesize,out_buffer,
112.         AV_PIX_FMT_YUV420P,pCodecCtx->width, pCodecCtx->height,1);
113.
114.
115.     packet=(AVPacket *)av_malloc(sizeof(AVPacket));
116.
117.     img_convert_ctx = sws_getContext(pCodecCtx->width, pCodecCtx->height, pCodecCtx->pix_fmt,
118.         pCodecCtx->width, pCodecCtx->height, AV_PIX_FMT_YUV420P, SWS_BICUBIC, NULL, NULL, NULL);
119.
120.
121.     sprintf(info, "[Input      ]%s\n", input_str);
122.     sprintf(info, "%s[Output    ]%s\n",info,output_str);
123.     sprintf(info, "%s[Format   ]%s\n",info, pFormatCtx->iformat->name);
124.     sprintf(info, "%s[Codec    ]%s\n",info, pCodecCtx->codec->name);
125.     sprintf(info, "%s[Resolution]%dx%d\n",info, pCodecCtx->width,pCodecCtx->height);
126.
127.
128.     fp_yuv=fopen(output_str,"wb+");
129.     if(fp_yuv==NULL){
130.         printf("Cannot open output file.\n");
131.         return -1;
132.     }
133.
134.     frame_cnt=0;
135.     time_start = clock();
136.
137.     while(av_read_frame(pFormatCtx, packet)>=0){
138.         if(packet->stream_index==videoindex){
139.             ret = avcodec_decode_video2(pCodecCtx, pFrame, &got_picture, packet);
140.             if(ret < 0){
141.                 LOGE("Decode Error.\n");
142.                 return -1;
143.             }
144.             if(got_picture){
145.                 sws_scale(img_convert_ctx, (const uint8_t* const*)pFrame->data, pFrame->linesize, 0, pCodecCtx->height,
146.                     pFrameYUV->data, pFrameYUV->linesize);
147.
148.                 y_size=pCodecCtx->width*pCodecCtx->height;
149.                 fwrite(pFrameYUV->data[0],1,y_size,fp_yuv);    //Y
150.                 fwrite(pFrameYUV->data[1],1,y_size/4,fp_yuv);  //U
151.                 fwrite(pFrameYUV->data[2],1,y_size/4,fp_yuv);  //V
152.                 //Output info
153.                 char pictype_str[10]={0};
154.                 switch(pFrame->pict_type){
155.                     case AV_PICTURE_TYPE_I:sprintf(pictype_str,"I");break;
156.                     case AV_PICTURE_TYPE_P:sprintf(pictype_str,"P");break;
157.                     case AV_PICTURE_TYPE_B:sprintf(pictype_str,"B");break;
158.                     default:sprintf(pictype_str,"Other");break;
159.                 }
160.                 LOGI("Frame Index: %5d. Type:%s",frame_cnt,pictype_str);
161.                 frame_cnt++;
162.             }
163.         }
164.         av_free_packet(packet);
165.     }
166.     //flush decoder
167.     //FIX: Flush Frames remained in Codec
168.     while (1) {
169.         ret = avcodec_decode_video2(pCodecCtx, pFrame, &got_picture, packet);
170.         if (ret < 0)
171.             break;
172.         if (!got_picture)
173.             break;
174.         sws_scale(img_convert_ctx, (const uint8_t* const*)pFrame->data, pFrame->linesize, 0, pCodecCtx->height,
175.             pFrameYUV->data, pFrameYUV->linesize);
176.         int y_size=pCodecCtx->width*pCodecCtx->height;
177.         fwrite(pFrameYUV->data[0],1,y_size,fp_yuv);    //Y
178.         fwrite(pFrameYUV->data[1],1,y_size/4,fp_yuv);  //U
179.         fwrite(pFrameYUV->data[2],1,y_size/4,fp_yuv);  //V

```

```

179.         twrite(prameruv->data[z],1,y_size/4,fp_yuv);  //v
180.         //Output info
181.         char pictype_str[10]={0};
182.         switch(pFrame->pict_type){
183.             case AV_PICTURE_TYPE_I:sprintf(pictype_str,"I");break;
184.             case AV_PICTURE_TYPE_P:sprintf(pictype_str,"P");break;
185.             case AV_PICTURE_TYPE_B:sprintf(pictype_str,"B");break;
186.             default:sprintf(pictype_str,"Other");break;
187.         }
188.         LOGI("Frame Index: %5d. Type:%s",frame_cnt,pictype_str);
189.         frame_cnt++;
190.     }
191.     time_finish = clock();
192.     time_duration=(double)(time_finish - time_start);
193.
194.     sprintf(info, "%s[Time      ]%fms\n",info,time_duration);
195.     sprintf(info, "%s[Count      ]%d\n",info,frame_cnt);
196.
197.     sws_freeContext(img_convert_ctx);
198.
199.     fclose(fp_yuv);
200.
201.     av_frame_free(&pFrameYUV);
202.     av_frame_free(&pFrame);
203.     avcodec_close(pCodecCtx);
204.     avformat_close_input(&pFormatCtx);
205.
206.     return 0;
207. }

```

Android.mk文件位于jni/Android.mk，如下所示。

```
[plain]
1. # Android.mk for FFmpeg
2. #
3. # Lei Xiaohua 雷霄骅
4. # leixiaohua1020@126.com
5. # http://blog.csdn.net/leixiaohua1020
6. #
7.
8. LOCAL_PATH := $(call my-dir)
9.
10. # FFmpeg library
11. include $(CLEAR_VARS)
12. LOCAL_MODULE := avcodec
13. LOCAL_SRC_FILES := libavcodec-56.so
14. include $(PREBUILT_SHARED_LIBRARY)
15.
16. include $(CLEAR_VARS)
17. LOCAL_MODULE := avdevice
18. LOCAL_SRC_FILES := libavdevice-56.so
19. include $(PREBUILT_SHARED_LIBRARY)
20.
21. include $(CLEAR_VARS)
22. LOCAL_MODULE := avfilter
23. LOCAL_SRC_FILES := libavfilter-5.so
24. include $(PREBUILT_SHARED_LIBRARY)
25.
26. include $(CLEAR_VARS)
27. LOCAL_MODULE := avformat
28. LOCAL_SRC_FILES := libavformat-56.so
29. include $(PREBUILT_SHARED_LIBRARY)
30.
31. include $(CLEAR_VARS)
32. LOCAL_MODULE := avutil
33. LOCAL_SRC_FILES := libavutil-54.so
34. include $(PREBUILT_SHARED_LIBRARY)
35.
36. include $(CLEAR_VARS)
37. LOCAL_MODULE := postproc
38. LOCAL_SRC_FILES := libpostproc-53.so
39. include $(PREBUILT_SHARED_LIBRARY)
40.
41. include $(CLEAR_VARS)
42. LOCAL_MODULE := swresample
43. LOCAL_SRC_FILES := libswresample-1.so
44. include $(PREBUILT_SHARED_LIBRARY)
45.
46. include $(CLEAR_VARS)
47. LOCAL_MODULE := swscale
48. LOCAL_SRC_FILES := libswscale-3.so
49. include $(PREBUILT_SHARED_LIBRARY)
50.
51. # Program
52. include $(CLEAR_VARS)
53. LOCAL_MODULE := sffdecoder
54. LOCAL_SRC_FILES :=simplest_ffmpeg_decoder.c
55. LOCAL_C_INCLUDES += $(LOCAL_PATH)/include
56. LOCAL_LDLIBS := -llog -lz
57. LOCAL_SHARED_LIBRARIES := avcodec avdevice avfilter avformat avutil postproc swresample swscale
58. include $(BUILD_SHARED_LIBRARY)
```

## 运行结果

App在手机上运行后的结果如下图所示。

□

注意需要把等待解码的视频文件拷贝至存储卡相应的目录中。例如对于上述截图的情况，需要将sintel.mp4拷贝至存储卡的根目录中。

单击“Start”按钮就可以将存储卡根目录中的视频文件解码为YUV文件（需要等待一段时间完成解码）。注意解码后的YUV文件体积巨大，可能会占用大量的存储卡空间。

□

## 下载

simplest ffmpeg mobile

项目主页

Github：[https://github.com/leixiaohua1020/simplest\\_ffmpeg\\_mobile](https://github.com/leixiaohua1020/simplest_ffmpeg_mobile)

开源中国：[https://git.oschina.net/leixiaohua1020/simplest\\_ffmpeg\\_mobile](https://git.oschina.net/leixiaohua1020/simplest_ffmpeg_mobile)

SourceForge：<https://sourceforge.net/projects/simplestffmpegmobile/>

CSDN工程下载地址：<http://download.csdn.net/detail/leixiaohua1020/8924391>

本解决方案包含了使用FFmpeg在移动端处理多媒体的各种例子：

[Android]

simplest\_android\_player: 基于安卓接口的视频播放器

simplest\_ffmpeg\_android\_helloworld: 安卓平台下基于FFmpeg的HelloWorld程序

**simplest\_ffmpeg\_android\_decoder: 安卓平台下最简单的基于FFmpeg的视频解码器**

simplest\_ffmpeg\_android\_decoder\_onelib: 安卓平台下最简单的基于FFmpeg的视频解码器-单库版

simplest\_ffmpeg\_android\_streamer: 安卓平台下最简单的基于FFmpeg的推流器

simplest\_ffmpeg\_android\_transcoder: 安卓平台下移植的FFmpeg命令行工具

simplest\_sdl\_android\_helloworld: 移植SDL到安卓平台的最简单程序

[IOS]

simplest\_ios\_player: 基于IOS接口的视频播放器

simplest\_ffmpeg\_ios\_helloworld: IOS平台下基于FFmpeg的HelloWorld程序

simplest\_ffmpeg\_ios\_decoder: IOS平台下最简单的基于FFmpeg的视频解码器

simplest\_ffmpeg\_ios\_streamer: IOS平台下最简单的基于FFmpeg的推流器

simplest\_ffmpeg\_ios\_transcoder: IOS平台下移植的ffmpeg.c命令行工具

simplest\_sdl\_ios\_helloworld: 移植SDL到IOS平台的最简单程序

版权声明：本文为博主原创文章，未经博主允许不得转载。 <https://blog.csdn.net/leixiaohua1020/article/details/47010637>

文章标签：

FFmpeg

Android

视频解码

JNI

个人分类：[Android多媒体](#)

[FFMPEG](#)

所属专栏：[FFmpeg](#)

此PDF由spygg生成, 请尊重原作者版权!!!

我的邮箱:liushidc@163.com