ffmpeg SDK□□□

□□□□□release□□

https://github.com/GyanD/codexffmpeg/releases

□□□□□dll□□□

https://github.com/donglietao/ffmpeg-dll

```
def ffmpeg():
    dll = ctypes.windll.LoadLibrary(
        "D:/cpp-sdk/FFmpeg-n7.0/ffmpeg-exe-dll/Release/release/ffmpeg-dll.dll"
    )

    ffmpeg_main = dll.ffmain

    ffmpeg_main.argtypes = [ctypes.c_int, ctypes.POINTER(ctypes.c_char_p)]
    ffmpeg_main.restype = ctypes.c_int
    argv = (ctypes.c_char_p * 10)()
    argv[0] = bytes("ffmpeg", "utf-8")
    argv[1] = bytes("-f", "utf-8")
    argv[2] = bytes("gdigrab", "utf-8")
    argv[3] = bytes("-i", "utf-8")
    argv[4] = bytes("desktop", "utf-8")
    argv[5] = bytes("-vcodec", "utf-8")
    argv[6] = bytes("libx264", "utf-8")
    argv[7] = bytes("-f", "utf-8")
    argv[8] = bytes("flv", "utf-8")
    argv[9] = bytes("rtmp://srs-push.qq829.cn:31935/live/557254322", "utf-8")
    ret = ffmpeg_main(10, argv)
    ctypes.windll.kernel32.FreeLibrary.argtypes = [ctypes.c_void_p]
    ctypes.windll.kernel32.FreeLibrary.restype = ctypes.c_int
    ctypes.windll.kernel32.FreeLibrary(dll._handle)
    return ret
```

```
ffmpeg()
```

ffmpeg编解码c++代码实现

```
// 注册输入和输出设备
avdevice_register_all();
// 初始化网络库
avformat_network_init();
//----------------打开输入设备----------------------//
// 确定输入格式
av_find_input_format("gdigrab");
// 打开输入文件
avformat_open_input();
// 查找输入流
avformat_find_stream_info();
// 选择流
in_stream = input_fmt_ctx->streams[i];
// 输入流的编解码器
AVCodec *in_codec = avcodec_find_decoder(in_stream->codecpar->codec_id);
// 解密上下文
in_codec_ctx = avcodec_alloc_context3(in_codec);
// 解码参数复制
avcodec_parameters_to_context();
// 打开解码器
avcodec_open2();
//----------------打开输处设备----------------------//
// 确定输出格式
av_guess_format()
// 构建输出上下文
avformat_alloc_output_context2();
// 添加输出流
AVStream *out_stream = avformat_new_stream(output_fmt_ctx, NULL);
// 设置流的参数
out_stream->codecpar->codec_id = AV_CODEC_ID_H264;
// 输出流编解码
AVCodec *out_codec = avcodec_find_encoder(out_stream->codecpar->codec_id);
// 输出流编码器上下文
```

```
AVCodecContext *out_codec_ctx = avcodec_alloc_context3(out_codec);
// 编码器参数复制
avcodec_parameters_to_context(out_codec_ctx, out_stream->codecpar);
// 打开编码器
avcodec_open2();
// 写入SPS和PPS信息
out_stream->codecpar->extradata = out_codec_ctx->extradata;
out_stream->codecpar->extradata_size = out_codec_ctx->extradata_size;
// 打开输出IO
avio_open();
// 写入文件头信息
avformat_write_header();
//----------------解密和转换----------------------//
while (1){
        // 从输入设备读取数据包
        ret = av_read_frame(input_fmt_ctx, &packet);
        // 解码数据包为帧
        ret = avcodec_send_packet(in_codec_ctx, &packet);
        // 从解码器中获取帧
        ret = avcodec_receive_frame(in_codec_ctx, frame);
        // 格式转换
        ret = sws_scale(sws_ctx, (const uint8_t *const *)frame->data, frame->linesize,
                        0, frame->height,
                        enc_frame->data, enc_frame->linesize);
        // 设置帧的时间戳
        enc_frame->pts = av_rescale_q_rnd((frame->pts - start_pts), in_stream->time_base,
out_stream->time_base, AV_ROUND_NEAR_INF | AV_ROUND_PASS_MINMAX);
        enc_frame->pkt_dts = enc_frame->pts;
        //   将帧发送到编码器
        ret = avcodec_send_frame(out_codec_ctx, enc_frame);
        // 从编码器中获取数据包
        while (avcodec_receive_packet(out_codec_ctx, &enc_packet) == 0)
        {
            // 写入数据包
            ret = av_interleaved_write_frame(output_fmt_ctx, &enc_packet);
            if (ret < 0)
            {
                printf("7:Error while writing output packet:%s\n", av_err2str(ret));
                break;
            }
        }
```

```
        // 释放编码数据包
        av_packet_unref(&enc_packet);
        // 释放编码数据包
        av_packet_unref(&packet);
}
//----------------后续工作----------------------//
    // 写入文件尾
    av_write_trailer(output_fmt_ctx);
    // 关闭输出文件
    avio_close(output_fmt_ctx->pb);
    // 释放数据包
    av_packet_free(&packet);
    av_packet_free(&enc_packet);
    // 释放帧
    av_frame_free(&frame);
    av_frame_free(&enc_frame);
    // 释放格式转换器
    sws_freeContext(sws_ctx);
    // 释放编码器上下文
    avcodec_free_context(&in_codec_ctx);
    avcodec_free_context(&out_codec_ctx);
    // 释放输入和输出设备
    avformat_close_input(&input_fmt_ctx);
    avformat_free_context(input_fmt_ctx);
    // 释放输出设备
    avformat_close_input(&output_fmt_ctx);
    avformat_free_context(output_fmt_ctx);
    // 释放网络库
    avformat_network_deinit();
```

版本号 #4
由 董列涛 创建于 1 六月 2024 17:42:46
由 董列涛 更新于 18 九月 2025 03:45:38