

## 原 最简单的基于FFmpeg的移动端例子：Windows Phone HelloWorld

2015年08月01日 22:25:48 阅读数：12904

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Windows Phone平台下基于FFmpeg的HelloWorld程序。该示例C语言的源代码来自于《[最简单的基于FFMPEG的Helloworld程序](#)》。相关的概念就不再重复记录了。

由于在FFmpeg移动端开发方面只有Android和IOS的实战经验，所以我一开始的时候只做了Android和IOS的示例程序。前两天要参加微软的Windows 10发布会，会前浏览信息的时候发现Windows 10在视音频处理方面已经加入了对FFmpeg的原生支持。同时微软还公布了一个开源项目FFmpegInterop，专门用于给Windows 8.1/10 App编译包含FFmpeg功能的类库。出于好奇我便下载并捣鼓了一下FFmpegInterop这个工程，最终总结了一个FFmpeg在Windows Phone平台的HelloWorld的示例程序。



## Windows Phone平台下使用FFmpeg类库的说明

Windows Phone平台（Windows App Store，Windows应用商店）使用FFmpeg类库的流程如下所示。

### 1. 编译FFmpeg类库

- （1）前提需要安装VS2013和MSYS2（最好已经可以成功使用这两个工具编译PC上使用的FFmpeg）。
- （2）获得FFmpegInterop工程（该工程位于Github上面，地址为 <https://github.com/Microsoft/FFmpegInterop>）。
- （3）加入FFmpeg源代码。从官网下载源代码后，将源代码解压到FFmpegInterop的ffmpeg目录下。

(4) 以管理员的身份启动“VS2013 开发人员命令提示”控制台，切换到FFmpegInterop目录。

(5) 配置MSYS2，运行下面命令设置MSYS2\_BIN环境变量（这里根据MSYS2的安装路径不同而不同）：

```
1. set MSYS2_BIN="E:\msys64\usr\bin\bash.exe"
```

(6) 处理一个小Bug。需要把MSYS2中的link.exe改个名字（随便改一个即可，例如“link\_msys.exe”）。由于MSYS2中的link.exe和VS2013中的link.exe重名了，如果不改名字的话系统就会错误地使用MSYS2的link.exe（而不是VS2013的link.exe），从而导致编译失败。不知道是不是所有的机器都有这个问题，当时确实卡了我一段时间。

(7) 执行命令编译类库。直接执行BuildFFmpeg.bat会打印帮助菜单。执行下面这条语句就可以编译Windows8.1 x86平台的类库。编译成功后的类库位于“ffmpeg\Build\Windows8.1\86”目录下。

```
1. BuildFFmpeg win8.1 x86
```

执行下面的语句会同时编译Windows8.1 x86和x64平台的类库。

```
1. BuildFFmpeg win8.1 x86 x64
```

其它的编译命令不再详述，可以查看帮助菜单。本步骤获得的FFmpeg类库就可以用于Windows App Store程序的开发了。

PS：这里生成的dll与平时用于控制台或者MFC程序中的dll是不一样的。这里的dll是加了AppContainer 的flag的dll。如果使用普通的控制台程序调用这里生成的dll，就会报错“0xc000a200”：

Error 0xc000a200: shows up when regular process (not inside an AppContainer) tries to load DLL that was marked with AppContainer flag。

(8) 打开samples文件夹下的sln解决方案[可选]。这一步sln解决方案中包含了FFmpegInterop库源代码工程以及一些示例程序。这部分的源代码还没有细看（本文暂不涉及这部分代码的内容）。

## 2. 编写Windows Phone平台下的程序

(1) 新建一个“Windows应用商店”程序。位于“文件->新建->项目->Visual C++->Windows应用商店->空白应用程序(XAML)”。

(2) 将编译后的类库拷贝至该项目目录下(注意x86、x64等这些平台要对应)。新建include文件夹存储头文件 (\*.h) ；lib文件夹存储导入库文件 (\*.lib) ；并将动态库文件 (\*.dll) 直接拷贝至目录下。

(3) 配置类库。分成以下3步：

### a) 头文件配置

解决方案资源管理器->右键单击项目->属性面板

属性面板->C/C++->常规->附加包含目录，输入“include”（刚才拷贝头文件的目录）

### b) 导入库配置

属性面板->链接器->常规->附加库目录，输入“lib”（刚才拷贝库文件的目录）

属性面板->链接器->输入->附加依赖项，输入“avcodec.lib; avformat.lib; avutil.lib; avdevice.lib; avfilter.lib; swresample.lib; swscale.lib”（导入库的文件名）

### c) 动态库配置（重要而有特色的一步）

解决方案资源管理器->右键单击项目->添加->现有项，将dll文件添加进来

选中每个dll文件->右键单击->属性->常规->内容，设定为“是”

(4) 测试

### a) 编辑界面

在MainPage.xaml中添加一个Button按钮，并添加一个“Button\_Clicked()”响应函数。

```
1. <Page
2.     x:Class="testApp.MainPage"
3.     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
4.     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
5.     xmlns:local="using:testApp"
6.     xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
7.     xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
8.     mc:Ignorable="d">
9.
10.     <StackPanel Margin="120,30,0,0">
11.         <TextBlock HorizontalAlignment="Left" Text="My FFmpeg test" FontSize="36"/>
12.         <Button Content="Configure Info" Click="Button_Clicked"/>
13.         <TextBlock x:Name="greetingOutput"/>
14.     </StackPanel>
15. </Page>
```

### b) 编辑代码

在MainPage.xaml.cpp中添加一个Button\_Clicked()的代码，如下所示。

```
[cpp]
1. void MainPage::Button_Clicked(Platform::Object^ sender, Windows::UI::Xaml::RoutedEventArgs^ e)
2. {
3.     //greetingOutput->Text = "Hello, Lei";
4.
5.     USES_CONVERSION;
6.     String ^info = ref new String(A2W(avcodec_configuration()));
7.     Windows::UI::Popups::MessageDialog(info).ShowAsync();
8. }
```

该代码调用了avcodec\_configuration()获取FFmpeg类库的配置信息，然后调用MessageDialog()弹出一个对话框显示这些信息。注意其中使用了atlconv.h中的一个A2W()的宏，用于把char \*转换为Platform::String。

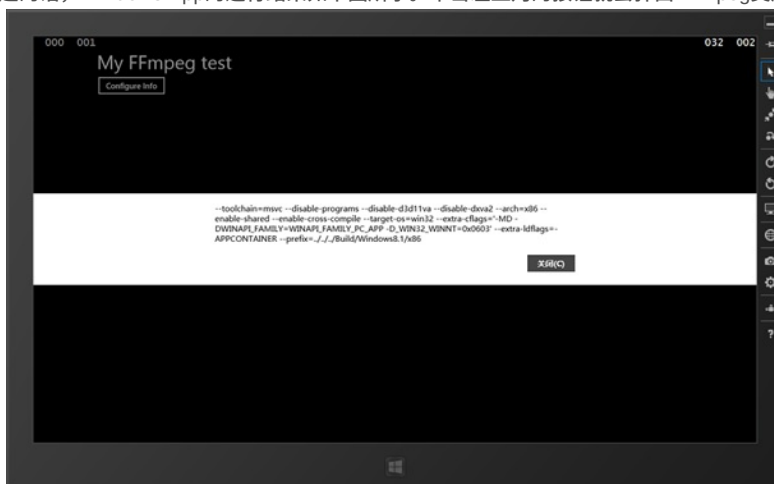
在MainPage.xaml.cpp头部添加用到的头文件，如下所示。

```
[cpp]
1. #include <atlconv.h>
2.
3. extern "C"{
4.     #include "libavcodec/avcodec.h"
5. }
```

在MainPage.xaml.h添加响应函数的声明，如下所示。

```
[cpp]
1. public ref class MainPage sealed
2. {
3. public:
4.     MainPage();
5.     void Button_Clicked(Platform::Object^ sender, Windows::UI::Xaml::RoutedEventArgs^ e);
6. };
```

如果一切配置都没有问题的话，Windows App的运行结果如下图所示。单击左上角的按钮就会弹出FFmpeg类库的配置信息。

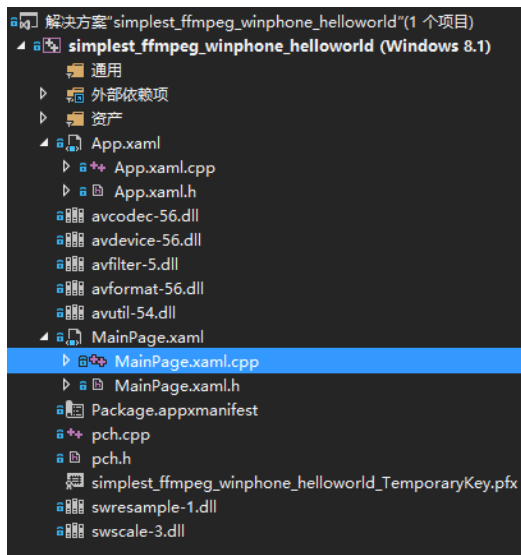


## (5) 其它资源

有关Windows App C++开发的详细资源可以参考MSDN上的《[使用 C++ 创建你的第一个通用 Windows 应用](#)》相关的文章。

## 源代码

Simplest FFmpeg WinPhone HelloWorld项目的目录结构如图所示。C++源代码位于MainPage.xaml.cpp中，界面布局文件为MainPage.xaml。



MainPage.xaml是界面布局文件，内容如下所示。

```
[html]
1. <!-- Simplest FFmpeg WinPhone HelloWorld -->
2. <Page
3.     x:Class="simplest_ffmpeg_winphone_helloworld.MainPage"
4.     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
5.     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
6.     xmlns:local="using:simplest_ffmpeg_winphone_helloworld"
7.     xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
8.     xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
9.     mc:Ignorable="d">
10.
11.     <StackPanel Margin="120,30,0,0">
12.         <TextBlock HorizontalAlignment="Left" Text="Simplest FFmpeg WinPhone HelloWorld" FontSize="36"/>
13.         <TextBlock Text="Click button to see FFmpeg lib's information"/>
14.         <StackPanel Orientation="Horizontal" Margin="0,20,0,20">
15.             <Button Content="Protocol" Click="click_protocol" Width="120" HorizontalAlignment="Left"/>
16.             <Button Content="AVFormat" Click="click_avformat" Width="120" HorizontalAlignment="Left"/>
17.             <Button Content="AVCodec" Click="click_avcodec" Width="120" HorizontalAlignment="Left"/>
18.             <Button Content="AVFilter" Click="click_avfilter" Width="120" HorizontalAlignment="Left"/>
19.             <Button Content="Configuration" Click="click_configuration" Width="120" HorizontalAlignment="Left"/>
20.         </StackPanel>
21.         <TextBlock x:Name="information" Margin="0,20,0,20"/>
22.     </StackPanel>
23. </Page>
```

MainPage.xaml.cpp是C++函数实现文件，内容如下所示。

```
[cpp]
1. /**
2.  * 最简单的Windows Phone平台下FFmpeg的HelloWorld例子
3.  * Simplest FFmpeg WinPhone HelloWorld
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.  * 本程序是移植FFmpeg到Windows App平台的最简单程序。它可以打印出FFmpeg类库的下列信息：
12.  * Protocol: FFmpeg类库支持的协议
13.  * AVFormat: FFmpeg类库支持的封装格式
14.  * AVCodec: FFmpeg类库支持的编解码器
15.  * AVFilter: FFmpeg类库支持的滤镜
16.  * Configure: FFmpeg类库的配置信息
17.  *
18.  * This is the simplest program based on FFmpeg in Windows App Platform. It can show following
19.  * informations about FFmpeg library:
20.  * Protocol: Protocols supported by FFmpeg.
21.  * AVFormat: Container format supported by FFmpeg.
22.  * AVCodec: Encoder/Decoder supported by FFmpeg.
23.  * AVFilter: Filters supported by FFmpeg.
24.  * Configure: configure information of FFmpeg.
25.  *
26.  */
27.
28. #include "pch.h"
29. #include "MainPage.xaml.h"
30. #include <atlconv.h>
31.
32. #define __STDC_CONSTANT_MACROS
33. extern "C"
```

```

34. {
35. #include "libavcodec/avcodec.h"
36. #include "libavformat/avformat.h"
37. #include "libavfilter/avfilter.h"
38. };
39.
40. using namespace simplest_ffmpeg_winphone_helloworld;
41.
42. using namespace Platform;
43. using namespace Windows::Foundation;
44. using namespace Windows::Foundation::Collections;
45. using namespace Windows::UI::Xaml;
46. using namespace Windows::UI::Xaml::Controls;
47. using namespace Windows::UI::Xaml::Controls::Primitives;
48. using namespace Windows::UI::Xaml::Data;
49. using namespace Windows::UI::Xaml::Input;
50. using namespace Windows::UI::Xaml::Media;
51. using namespace Windows::UI::Xaml::Navigation;
52. using namespace Windows::UI::Popups;
53.
54. MainPage::MainPage()
55. {
56.     InitializeComponent();
57. }
58.
59. /**
60.  * Protocol Support Information
61.  */
62. void MainPage::click_protocol(Platform::Object^ sender, Windows::UI::Xaml::RoutedEventArgs^ e){
63.
64.     //FIX
65.     struct URLProtocol;
66.
67.     char info[40000] = {0};
68.     av_register_all();
69.
70.     struct URLProtocol *pup = NULL;
71.     //Input
72.     struct URLProtocol **p_temp = &pup;
73.     avio_enum_protocols((void **)p_temp, 0);
74.     while ((*p_temp) != NULL){
75.         sprintf_s(info, sizeof(info), "%s[In ]%10s\n", info, avio_enum_protocols((void **)p_temp, 0));
76.     }
77.     pup = NULL;
78.     //Output
79.     avio_enum_protocols((void **)p_temp, 1);
80.     while ((*p_temp) != NULL){
81.         sprintf_s(info, sizeof(info), "%s[Out]%10s\n", info, avio_enum_protocols((void **)p_temp, 1));
82.     }
83.
84.     USES_CONVERSION;
85.     String ^infostr = ref new String(A2W(info));
86.     information->Text = infostr;
87.
88. }
89.
90. /**
91.  * AVFormat Support Information
92.  */
93.
94. void MainPage::click_avformat(Platform::Object^ sender, Windows::UI::Xaml::RoutedEventArgs^ e){
95.
96.     char info[40000] = { 0 };
97.
98.     av_register_all();
99.
100.     AVInputFormat *if_temp = av_iformat_next(NULL);
101.     AVOutputFormat *of_temp = av_oformat_next(NULL);
102.     //Input
103.     while (if_temp != NULL){
104.         sprintf_s(info, sizeof(info), "%s[In ] %10s\n", info, if_temp->name);
105.         if_temp = if_temp->next;
106.     }
107.     //Output
108.     while (of_temp != NULL){
109.         sprintf_s(info, sizeof(info), "%s[Out] %10s\n", info, of_temp->name);
110.         of_temp = of_temp->next;
111.     }
112.
113.     USES_CONVERSION;
114.     String ^infostr = ref new String(A2W(info));
115.     information->Text = infostr;
116.
117. }
118.
119. /**
120.  * AVCodec Support Information
121.  */
122. void MainPage::click_avcodec(Platform::Object^ sender, Windows::UI::Xaml::RoutedEventArgs^ e){
123.
124.     char info[40000] = { 0 };

```

```

125.
126.     av_register_all();
127.
128.     AVCodec *c_temp = av_codec_next(NULL);
129.
130.     while (c_temp != NULL){
131.         if (c_temp->decode != NULL){
132.             sprintf_s(info, sizeof(info), "%s[Dec]", info);
133.         }
134.         else{
135.             sprintf_s(info, sizeof(info), "%s[Enc]", info);
136.         }
137.         switch (c_temp->type){
138.             case AVMEDIA_TYPE_VIDEO:
139.                 sprintf_s(info, sizeof(info), "%s[Video]", info);
140.                 break;
141.             case AVMEDIA_TYPE_AUDIO:
142.                 sprintf_s(info, sizeof(info), "%s[Audio]", info);
143.                 break;
144.             default:
145.                 sprintf_s(info, sizeof(info), "%s[Other]", info);
146.                 break;
147.         }
148.
149.         sprintf_s(info, sizeof(info), "%s %10s\n", info, c_temp->name);
150.
151.         c_temp = c_temp->next;
152.     }
153.
154.     USES_CONVERSION;
155.     String ^infostr = ref new String(A2W(info));
156.     information->Text = infostr;
157.
158. }
159.
160. /**
161.  * AVFilter Support Information
162.  */
163. void MainPage::click_avfilter(Platform::Object^ sender, Windows::UI::Xaml::RoutedEventArgs^ e){
164.
165.     char info[40000] = { 0 };
166.     avfilter_register_all();
167.     AVFilter *f_temp = (AVFilter *)avfilter_next(NULL);
168.     while (f_temp != NULL){
169.         sprintf_s(info, sizeof(info), "%s[%10s]\n", info, f_temp->name);
170.     }
171.
172.     USES_CONVERSION;
173.     String ^infostr = ref new String(A2W(info));
174.     information->Text = infostr;
175.
176. }
177.
178. /**
179.  * Configuration Information
180.  */
181. void MainPage::click_configuration(Platform::Object^ sender, Windows::UI::Xaml::RoutedEventArgs^ e){
182.
183.     char info[10000] = { 0 };
184.     av_register_all();
185.
186.     sprintf_s(info, sizeof(info), "%s\n", avcodec_configuration());
187.
188.     USES_CONVERSION;
189.     String ^infostr = ref new String(A2W(avcodec_configuration()));
190.     //information->Text = infostr;
191.
192.     MessageDialog(infostr).ShowAsync();
193.
194. }

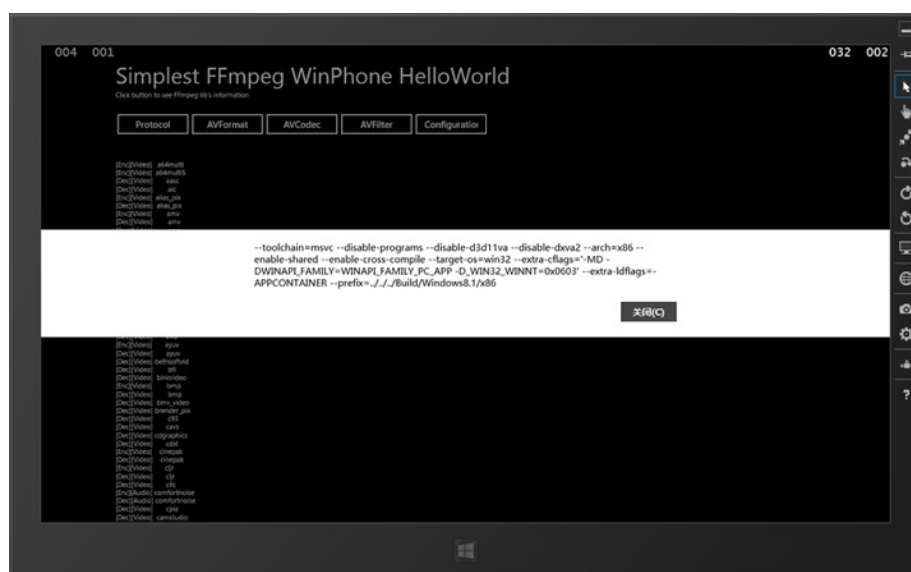
```

## 运行结果

程序运行后界面如下图所示。单击不同的按钮会显示FFmpeg类库不同方面的信息。



单击“Configure”按钮的时候会以消息框的形式打印配置信息。



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/>

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

[Windows]

simplest\_ffmpeg\_windowsphone\_helloworld: Windows Phone平台下基于FFmpeg的HelloWorld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/47191283>

文章标签：[FFmpeg](#) [Windows](#) [编译](#)

个人分类：[FFMPEG](#) [WinPhone多媒体](#)

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