

# Windows 10开发环境搭建

---

## 1. FFmpeg命令行环境搭建

[下载4.2.1版本源码](#)

[下载4.2.1编译好的文件](#)

[FFmpeg命令行环境搭建](#)

## 2. VS2015安装

## 3. QT 5.10安装

[3.1 安装CDB调试器步骤\(MSVC编译器调试用\)](#)

[3.2 直接使用FFmpeg官方编译的库](#)

[引用FFMPEG库](#)

[修改main.c文件](#)

[执行程序](#)

## 4. Windows编译FFmpeg

[4.1 编译环境](#)

[4.2 修改pacman的源](#)

[4.3 安装编译环境](#)

[4.3.1安装mingw-w64](#)

[4.3.2 安装git](#)

[4.3.3 安装make等工具](#)

[4.4 编译环境的其他准备工作](#)

[4.5 编译第三方库](#)

[4.5.1 下载和编译x264](#)

[4.5.2 下载和编译fdk-aac](#)

[4.5.3 下载编译mp3](#)

[4.5.4 下载编译libvpx](#)

[4.6 下载和编译ffmpeg](#)

[4.7 MinGW与MSVC编译的区别](#)

## 5 QT使用MSVC2015 64bit调用我们编译的ffmpeg库

腾讯课堂 零声学院

FFmpeg/WebRTC/RTMP音视频流媒体高级开发 <https://ke.qq.com/course/468797?tuin=137bb271>

Darren QQ 326873713

开发环境 Win10

## 1. FFmpeg命令行环境搭建

FFMPEG官网: <http://ffmpeg.org/>

### 下载4.2.1版本源码

源码: <https://ffmpeg.org/releases/ffmpeg-4.2.1.tar.bz2>

### 下载4.2.1编译好的文件

下载已经编译好的FFMPEG

网址: <https://ffmpeg.zeranoe.com/builds/>

Version	Architecture	Linking
20191212-f58bda6	Windows 64-bit	Static
4.2.1	Windows 32-bit	Shared
	macOS 64-bit	Dev

Download Build

32位下载地址:

Shared: 包含FFMPEG的dll库文件

<https://ffmpeg.zeranoe.com/builds/win32/shared/ffmpeg-4.2.1-win32-shared.zip>

Static: 包含了FFMPEG的官方文档

<https://ffmpeg.zeranoe.com/builds/win32/static/ffmpeg-4.2.1-win32-static.zip>

Dev: 包含FFMPEG的lib文件/头文件, 以及example范例。

<https://ffmpeg.zeranoe.com/builds/win32/dev/ffmpeg-4.2.1-win32-dev.zip>

我们目前主要是使用32位的版本。

## FFmpeg命令行环境搭建

解压ffmpeg-4.2.1-win32-shared.zip

### 1. 拷贝可执行文件到C:\Windows

media > src > 06-ffmpeg\_basic > ffmpeg-4.2.1-win32-shared > bin

名称	修改日期	类型	大小
ffmpeg.exe	2019/9/15 22:17	应用程序	287 KB
ffplay.exe	2019/9/15 22:17	应用程序	145 KB
ffprobe.exe	2019/9/15 22:17	应用程序	163 KB
avcodec-58.dll	2019/9/15 22:17	应用程序扩展	33,443 KB
avdevice-58.dll	2019/9/15 22:17	应用程序扩展	1,406 KB
avfilter-7.dll	2019/9/15 22:17	应用程序扩展	7,194 KB
avformat-58.dll	2019/9/15 22:17	应用程序扩展	9,672 KB
avutil-56.dll	2019/9/15 22:17	应用程序扩展	763 KB
postproc-55.dll	2019/9/15 22:17	应用程序扩展	122 KB
swresample-3.dll	2019/9/15 22:17	应用程序扩展	310 KB
swscale-5.dll	2019/9/15 22:17	应用程序扩展	507 KB

### 2. 拷贝动态链接库到C:\Windows\SysWOW64

(WoW64 (Windows On Windows64 <sup>[1]</sup>) 是一个Windows操作系统的子系统，被设计用来处理许多在32-bit Windows和64-bit Windows之间的不同的问题，使得可以在64-bit Windows中运行32-bit程序。)

media > src > 06-ffmpeg\_basic > ffmpeg-4.2.1-win32-shared > bin

名称	修改日期	类型	大小
swscale-5.dll	2019/9/15 22:17	应用程序扩展	507 KB
swresample-3.dll	2019/9/15 22:17	应用程序扩展	310 KB
postproc-55.dll	2019/9/15 22:17	应用程序扩展	122 KB
avutil-56.dll	2019/9/15 22:17	应用程序扩展	763 KB
avformat-58.dll	2019/9/15 22:17	应用程序扩展	9,672 KB
avfilter-7.dll	2019/9/15 22:17	应用程序扩展	7,194 KB
avdevice-58.dll	2019/9/15 22:17	应用程序扩展	1,406 KB
avcodec-58.dll	2019/9/15 22:17	应用程序扩展	33,443 KB
ffprobe.exe	2019/9/15 22:17	应用程序	163 KB
ffplay.exe	2019/9/15 22:17	应用程序	145 KB
ffmpeg.exe	2019/9/15 22:17	应用程序	287 KB

### 3. 打开cmd命令行窗口

输入ffmpeg -version测试，打印版本号4.2.1即可。

```

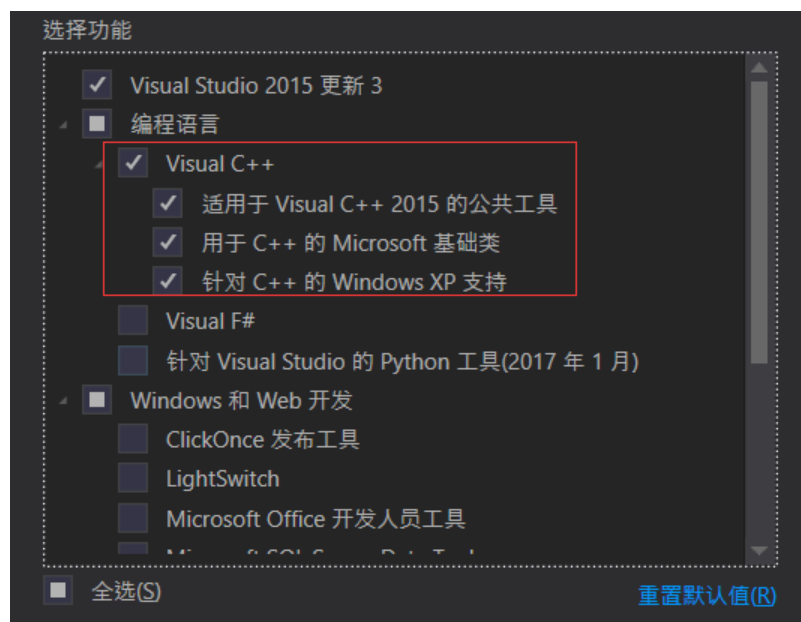
C:\Users\MUYUY>ffmpeg -version
ffmpeg version 4.2.1 Copyright (c) 2000-2019 the FFmpeg developers
built with gcc 9.1.1 (GCC) 20190807
configuration: --disable-static --enable-shared --enable-gpl --enable-version3 --en
e-gnutls --enable-iconv --enable-libass --enable-libdav1d --enable-libbluray --en
enable-libbrotli --enable-libbrotli --enable-libbrotli --enable-libbrotli --enable
enable-libbrotli --enable-libbrotli --enable-libbrotli --enable-libbrotli --enable
snappy --enable-libsoxr --enable-libtheora --enable-libtwolame --enable-libvpx --e
able-libx264 --enable-libx265 --enable-libx265 --enable-libx265 --enable-libx265 --e
stab --enable-libvorbis --enable-libvo-amrwbenc --enable-libmysofa --enable-libsp
enable-libmfx --enable-amf --enable-ffnvcodec --enable-cuvid --enable-d3d11va --e
a2 --enable-avisynth --enable-libopenmpt
libavutil 56. 31.100 / 56. 31.100
libavcodec 58. 54.100 / 58. 54.100
libavformat 58. 29.100 / 58. 29.100
libavdevice 58.  9.100 / 58.  9.100
libavfilter 7. 57.100 / 7. 57.100
libswscale 5.  5.100 / 5.  5.100
libswresample 35.  3.100 / 35.  3.100
libpostproc 55.  1.100 / 55.  1.100

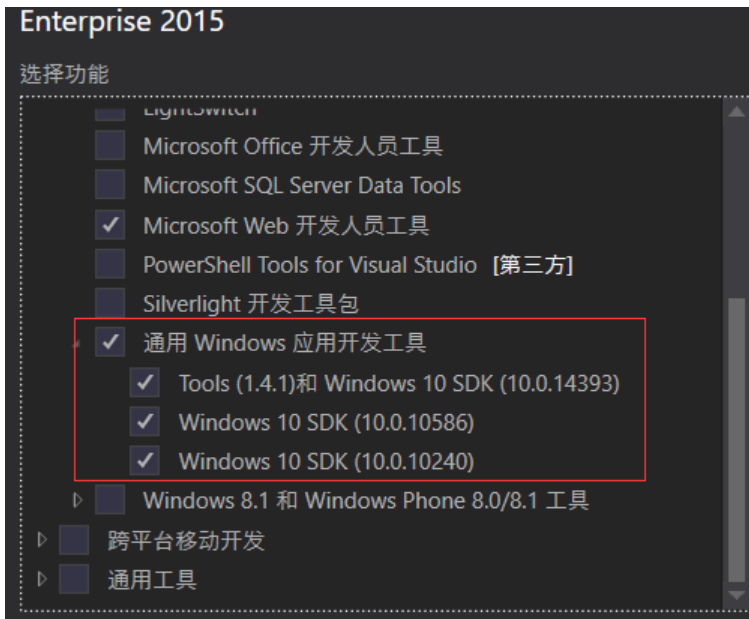
```

对于命令行实战中遇到FFmpeg版本号不同的时候不用担心，之前录制FFmpeg命令行实战的时候最新的版本是4.1，现在用4.2.1版本测试是没有任何问题。

## 2. VS2015安装

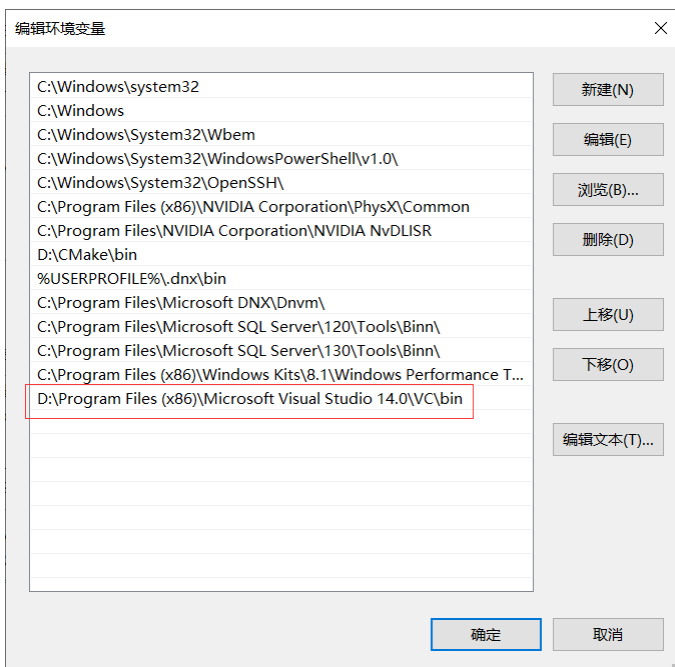
第三方下载：<http://www.xz7.com/downinfo/202013.html>





Enterprise: HM6NR-QXX7C-DFW2Y-8B82K-WTYJV

设置环境变量



### 3. QT 5.10安装

下载版本: QT版本 5.10.1

下载地址: [http://download.qt.io/official\\_releases/qt/5.10/5.10.1/](http://download.qt.io/official_releases/qt/5.10/5.10.1/)

选择该版本

Name	Last modified	Size	Metadata
↑ Parent Directory		-	
submodules/	11-Feb-2018 10:55	-	
single/	11-Feb-2018 11:07	-	
qt-opensource-windows-x86-pdb-files-uwp-5.10.1.zip	12-Feb-2018 10:22	1.4G	Details
qt-opensource-windows-x86-pdb-files-desktop-5.10.1.zip	12-Feb-2018 10:18	1.8G	Details
qt-opensource-windows-x86-5.10.1.exe	11-Feb-2018 11:19	2.3G	Details
qt-opensource-mac-x64-5.10.1.dmg	11-Feb-2018 11:15	2.5G	Details
qt-opensource-linux-x64-5.10.1.run	11-Feb-2018 11:10	1.0G	Details
md5sums.txt	12-Feb-2018 15:06	381	Details

直接下载地址: <http://iso.mirrors.ustc.edu.cn/qtproject/archive/qt/5.10/5.10.1/qt-opensource-windows-x86-5.10.1.exe>

下载地址:

```
No winrunner.exe found.
Project ERROR: Cannot run compiler 'cl'. Output:
=====
=====
Maybe you forgot to setup the environment?
Error while parsing file C:\Users\32687\Documents\qt-c\qt-c.pro. Giving up.
Project ERROR: Cannot run compiler 'cl'. Output:
=====
=====
Maybe you forgot to setup the environment?
Error while parsing file D:\0voice\av_media\qt\qt-c\qt-c.pro. Giving up.
```

```
1 Running Windows Runtime device detection.
2 No winrunner.exe found.
3 Project ERROR: Cannot run compiler 'cl'. Output:
4 =====
5 =====
6 Maybe you forgot to setup the environment?
7 Error while parsing file C:\Users\32687\Documents\qt-c\qt-c.pro.
  Giving up.
8 Project ERROR: Cannot run compiler 'cl'. Output:
9 =====
10 =====
11 Maybe you forgot to setup the environment?
12 Error while parsing file D:\0voice\av_media\qt\qt-c\qt-c.pro. Giv
    ing up.
```

## 3.1 安装CDB调试器步骤(MSVC编译器调试用)

先关闭Qt Creator

msvc编译器使用windbg下的cdb调试器 所以需要安装windbg

官网下载链接：<https://docs.microsoft.com/en-us/windows-hardware/drivers/debugger/debugger-download-tools>

Qt Creator 安装SDK，在MSVC编译模式下使用CDB调试器

<https://www.cnblogs.com/lixuejian/p/12915174.html>

### 即刻体验

可以通过两种方式下载 Windows 10 SDK：选择本页上的下载

安装此 SDK 之前：

1. 查看所有 [系统要求](#)
2. 请在安装前退出 Visual Studio 2019。
3. 检查 [发行说明和已知问题](#)。

下载安装程序 >

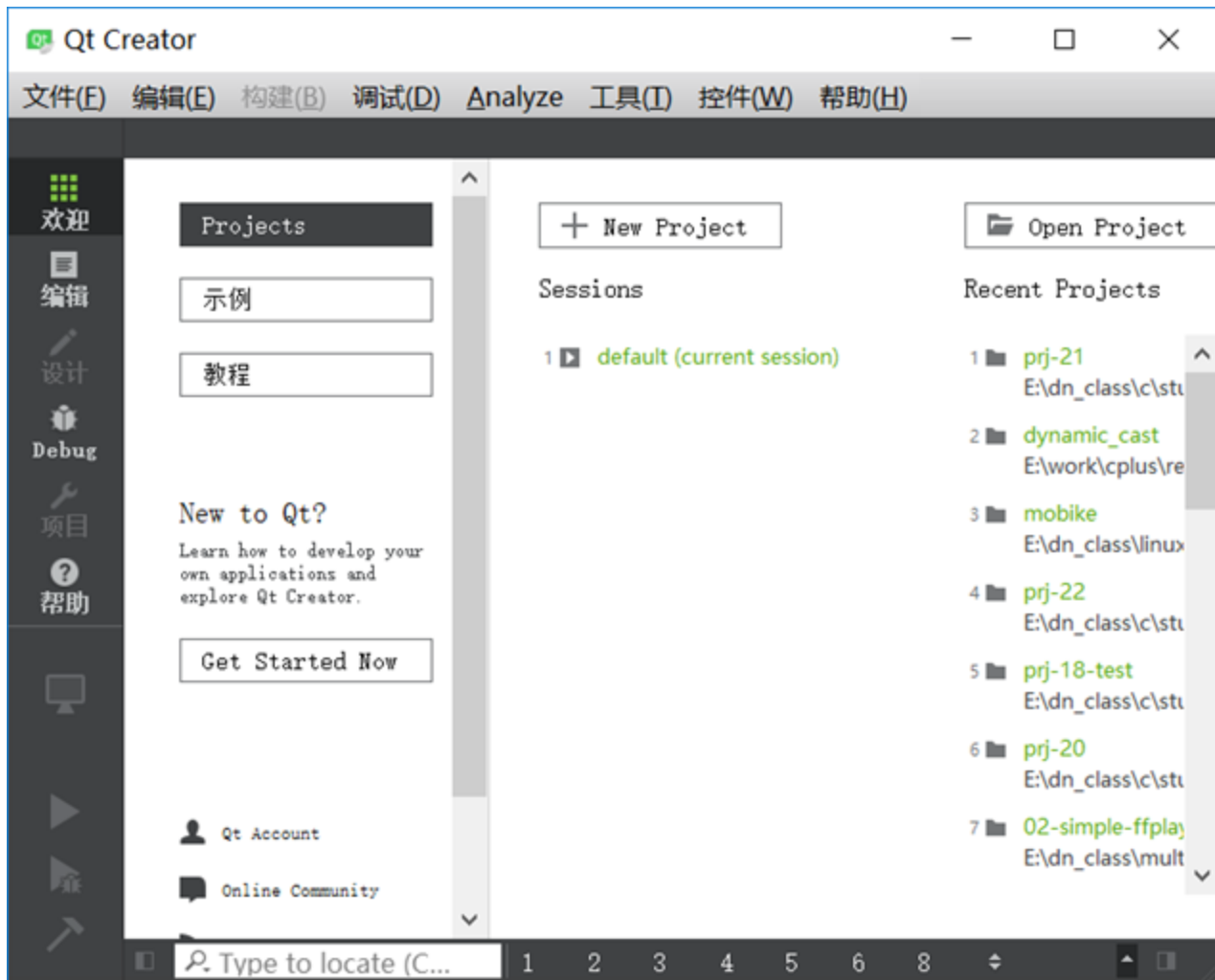
[下载 .ISO >](#)

更新时间：20/12/16

一定要安装好该调试器才能 MSVC2015 32bit/64bit才能调试。

## 3.2 直接使用FFmpeg官方编译的库

刚打开QT Creator的界面

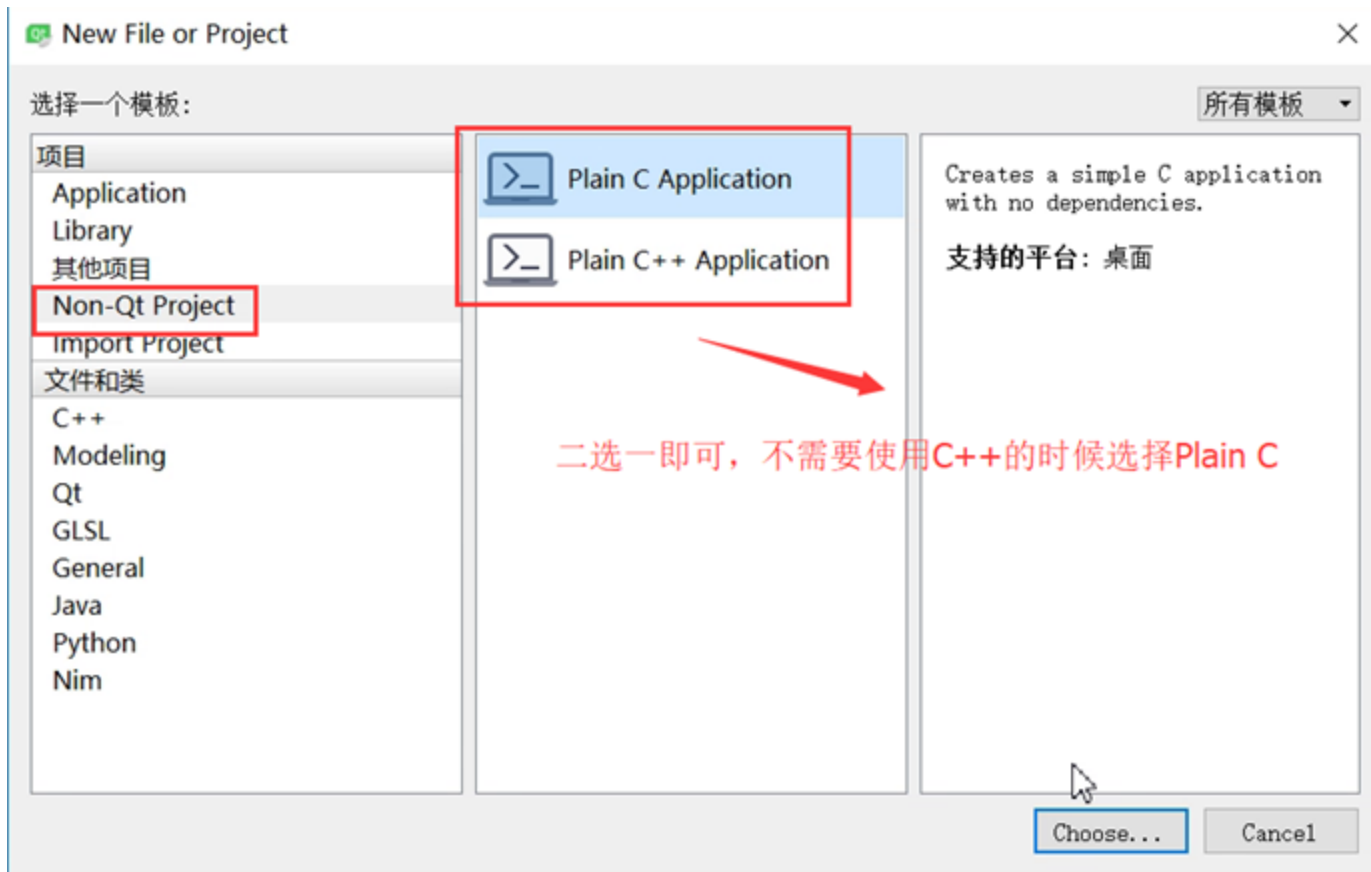


## 2 新建工程

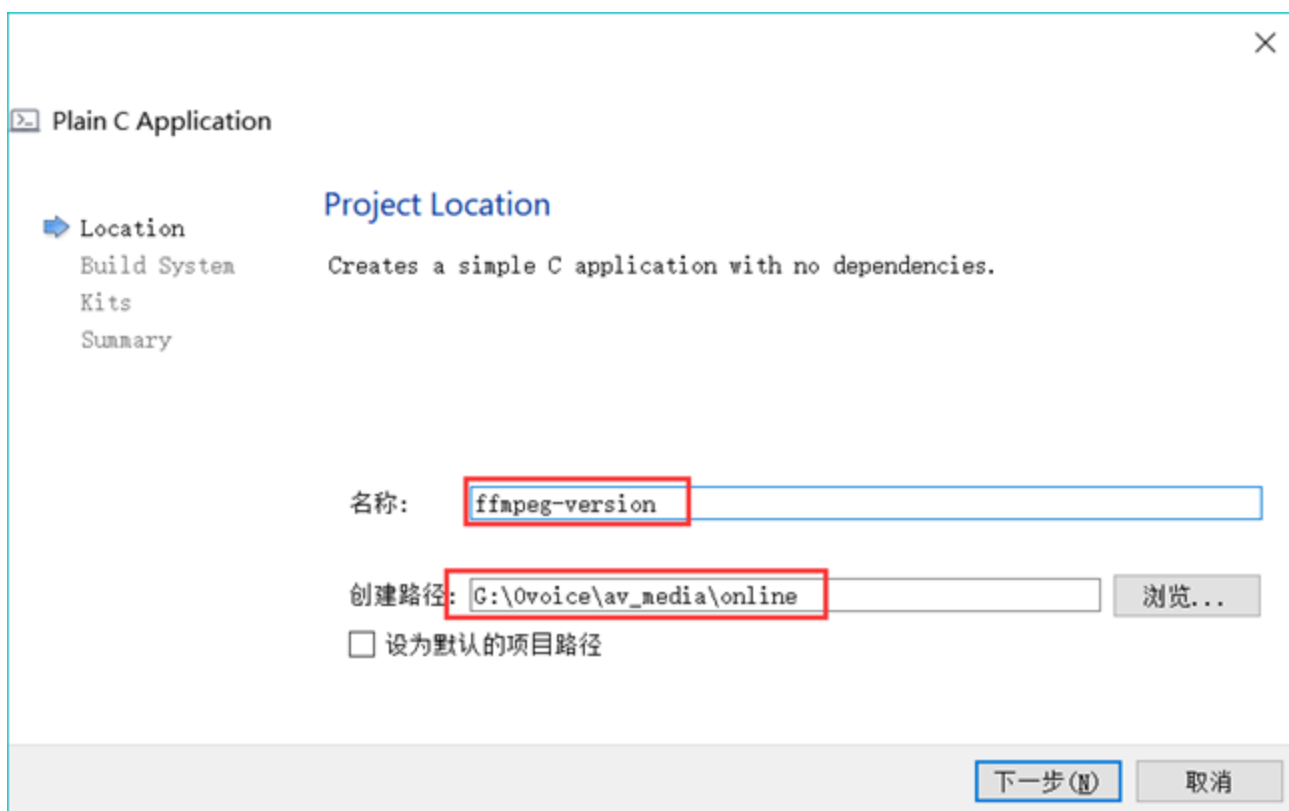




3 选择Non-Qt Project



4 填写项目名称以及路径，如下所示就创建了一个叫ffmpeg-version的工程。





← Plain C Application

## Define Build System

Location

➔ Build System

Kits

Summary

Build system: qmake

下一步(N)

取消



← Plain C Application

## Kit Selection

Location

Build System

➔ Kits

Summary

The following kits can be used for project **ffmpeg-version**:

☒ Select all kits

<input type="checkbox"/>	Desktop Qt 5.10.1 MSVC2015 32bit	详情
<input type="checkbox"/>	Desktop Qt 5.10.1 MSVC2015 64bit	详情
<input type="checkbox"/>	Desktop Qt 5.10.1 MSVC2017 64bit	详情
<input checked="" type="checkbox"/>	Desktop Qt 5.10.1 MinGW 32bit	详情

选择该编译器

下一步(N)

取消

← Plain C Application

## Project Management

Location

Build System

Kits

➔ Summary

作为子项目添加到项目中:

&lt;None&gt;

添加到版本控制系统(V):

&lt;None&gt;

Configure...

要添加的文件

E:\dn\_class\multimedia\src\01-code\ffmpeg-version:

ffmpeg-version.pro  
main.c

完成(F)

取消

到此创建了一个基本的工程。

注意：需要使用C++时则选择



Plain C++ Application。

## 引用FFMPEG库

将ffmpeg-4.2.1-win32-dev拷贝到ffmpeg-version目录下

此电脑 > 本地磁盘 (G:) > Ovoice > av_media > online > ffmpeg-version					搜索"ffr
名称	修改日期	类型	大小		
ffmpeg-4.2.1-win32-dev	2019/12/14 17:45	文件夹			
ffmpeg-version.pro	2019/12/14 17:45	Qt Project file	1 KB		
ffmpeg-version.pro.user	2019/12/14 17:42	Visual Studio Pr...	23 KB		
main.c	2019/12/14 17:42	C Source file	1 KB		

在ffmpeg-version.pro里面添加ffmpeg头文件和库文件路径

```
1 TEMPLATE = app
2 CONFIG += console
3 CONFIG -= app_bundle
4 CONFIG -= qt
5
6 SOURCES += main.c
7 win32 {
8 INCLUDEPATH += $$PWD/ffmpeg-4.2.1-win32-dev/include
9 LIBS += $$PWD/ffmpeg-4.2.1-win32-dev/lib/avformat.lib \
10         $$PWD/ffmpeg-4.2.1-win32-dev/lib/avcodec.lib \
11         $$PWD/ffmpeg-4.2.1-win32-dev/lib/avdevice.lib \
12         $$PWD/ffmpeg-4.2.1-win32-dev/lib/avfilter.lib \
13         $$PWD/ffmpeg-4.2.1-win32-dev/lib/avutil.lib \
14         $$PWD/ffmpeg-4.2.1-win32-dev/lib/postproc.lib \
15         $$PWD/ffmpeg-4.2.1-win32-dev/lib/swresample.lib \
16         $$PWD/ffmpeg-4.2.1-win32-dev/lib/swscale.lib
17 }
```

即是

```
win32 {
INCLUDEPATH += $$PWD/ffmpeg-4.2.1-win32-dev/include
LIBS += $$PWD/ffmpeg-4.2.1-win32-dev/lib/avformat.lib \
        $$PWD/ffmpeg-4.2.1-win32-dev/lib/avcodec.lib \
        $$PWD/ffmpeg-4.2.1-win32-dev/lib/avdevice.lib \
        $$PWD/ffmpeg-4.2.1-win32-dev/lib/avfilter.lib \
        $$PWD/ffmpeg-4.2.1-win32-dev/lib/avutil.lib \
        $$PWD/ffmpeg-4.2.1-win32-dev/lib/postproc.lib \
        $$PWD/ffmpeg-4.2.1-win32-dev/lib/swresample.lib \
        $$PWD/ffmpeg-4.2.1-win32-dev/lib/swscale.lib
}
```

LIBS的多行引用一定要记得带斜杠，否则后续的引用无效。

## 修改main.c文件

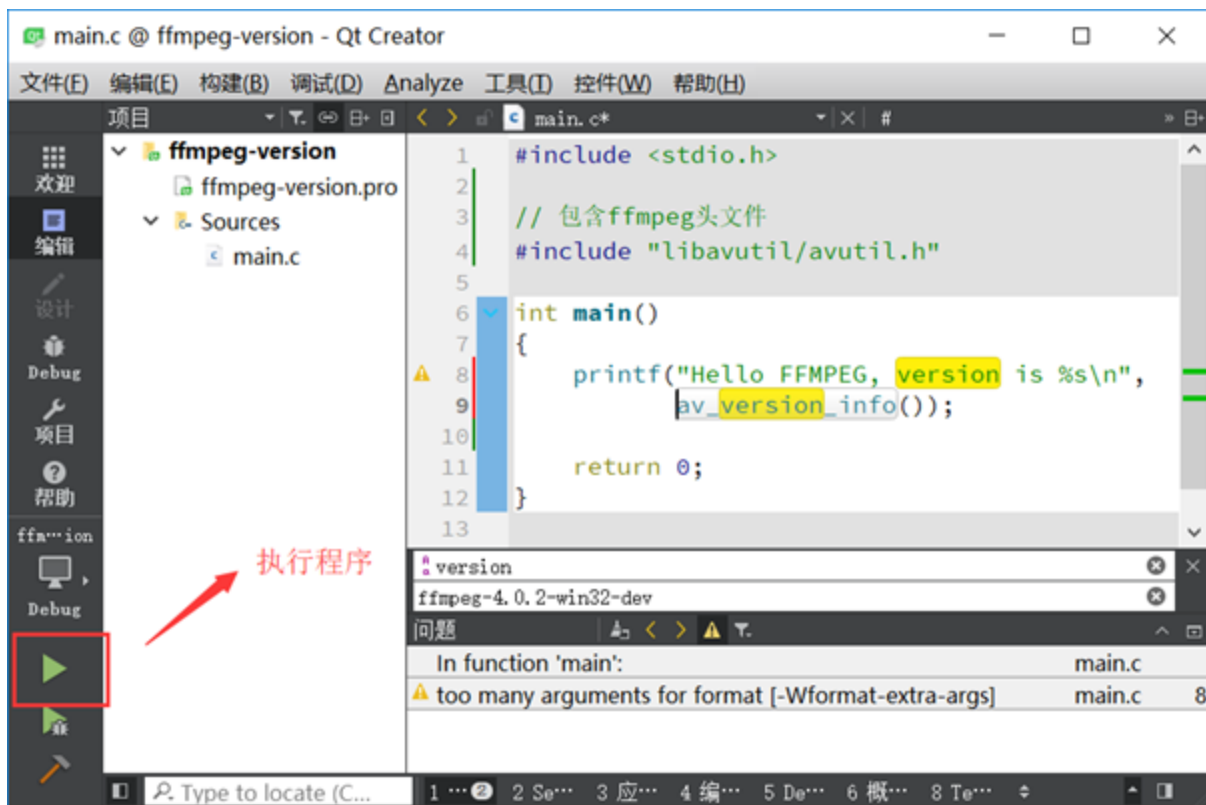
```
#include <stdio.h>

// 包含ffmpeg头文件
#include "libavutil/avutil.h"
```

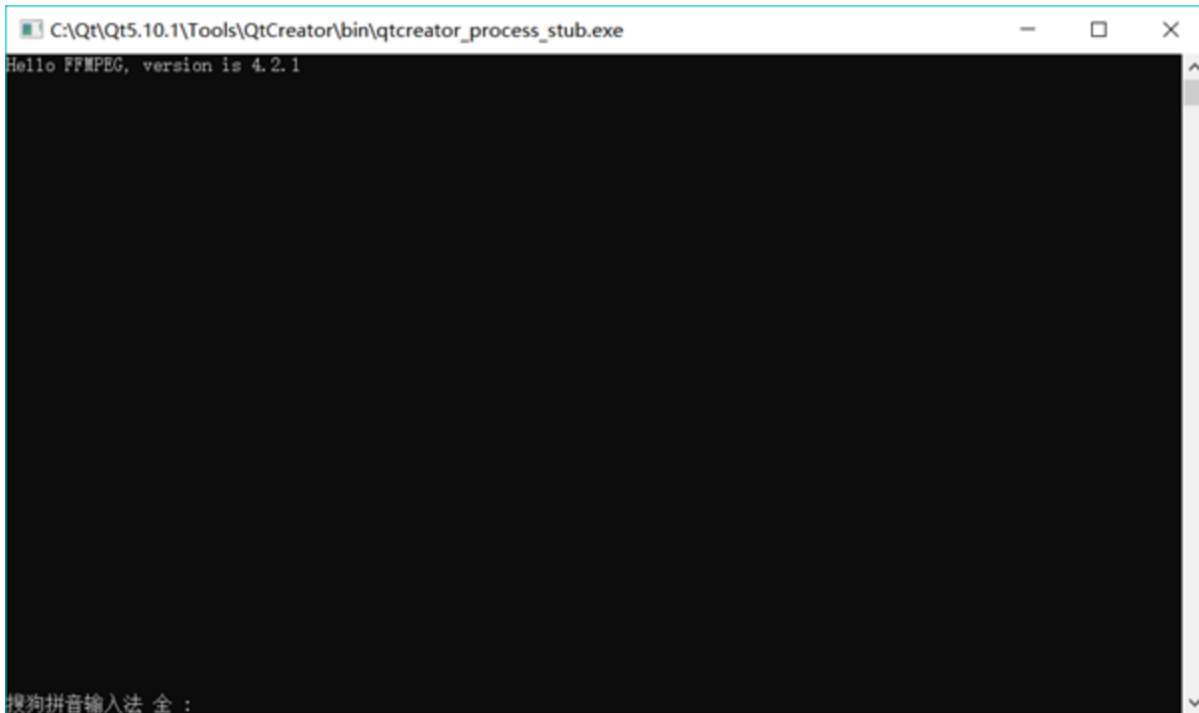
```
int main()
{
    printf("Hello FFmpeg, version is %s\n", av_version_info());

    return 0;
}
```

## 执行程序



显示 Hello FFmpeg, version is 4.2.1



## 4. Windows编译FFmpeg

### 4.1 编译环境

MSYS2 (Minimal SYStem 2)是一个MSYS的独立改写版本，主要用于 shell 命令行开发环境。同时它也是一个在Cygwin(POSIX 兼容性层) 和 MinGW-w64(从"MinGW-生成")基础上产生的，追求更好的互操作性的 Windows 软件。

MSYS2 是MSYS的一个升级版,准确的说是集成了panman和Mingw-64的Cygwin升级版, 提供了bash shell等linux环境、版本控制软件 (git/hg) 和MinGW-w64 工具链。与MSYS最大的区别是移植了Arch linux的软件包管理系统Pacman(其实是与Cygwin的区别)。

1. 下载安装MSYS2(按照官网安装到自己指定的目录下，本人安装于D:/msys64)

2. 安装完成之后,先把安装目录下的msys2\_shell.cmd中注释掉的 `rem set`

`MSYS2_PATH_TYPE=inherit`改成 `set MSYS2_PATH_TYPE=inherit`，这是为了将vs的环境继承给MSYS2。

```
16 rem or uncomment next line
17 set MSYS2_PATH_TYPE=inherit
18
```

MSYS2可以选择msys或者MinGW-w64环境来编译，不过在msys下使用gcc编译出来的exe和dll依赖 `msys-2.0.dll`，而MinGW-w64下编译出来的文件不需要依赖这个dll，从程序的运行效率来看，不依赖这个dll的程序效率应该更高。所以选择MinGW-w64来编译更佳。

下载地址：<https://www.msys2.org/>

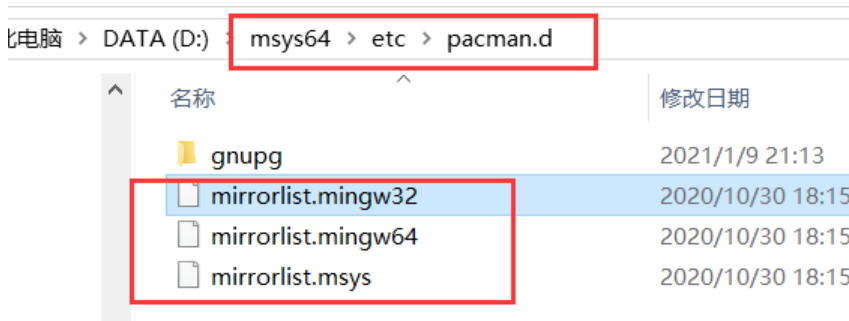
## Installation

1. Download the installer: [msys2-x86\\_64-20201109.exe](#)

## 4.2 修改pacman的源

pacman是一个软件包管理器，用来在MSYS2中安装软件，但是默认的国外的源下载安装包时非常缓慢，大概只有十几二十KB的速度，而且还容易下载中断出错，所以需要修改为国内源，国内源可以选择中科大的源。

按照MSYS2镜像提示修改。具体如下：



注意：是在文件夹打开文件进行编辑，不是在shell窗口编辑。

编辑 `/etc/pacman.d/mirrorlist.mingw32`，在文件开头添加：

```
1 Server = https://mirrors.tuna.tsinghua.edu.cn/msys2/mingw/i686/
2 Server = http://mirrors.ustc.edu.cn/msys2/mingw/i686/
```

编辑 `/etc/pacman.d/mirrorlist.mingw64`，在文件开头添加：

```
1 Server = https://mirrors.tuna.tsinghua.edu.cn/msys2/mingw/x86_64/
2 Server = http://mirrors.ustc.edu.cn/msys2/mingw/x86_64/
```

编辑 `/etc/pacman.d/mirrorlist.msys`，在文件开头添加：

```
1 Server = https://mirrors.tuna.tsinghua.edu.cn/msys2/msys/$arch/
2 Server = http://mirrors.ustc.edu.cn/msys2/msys/$arch/
```

3.启动命令行窗口，在窗口中输入：

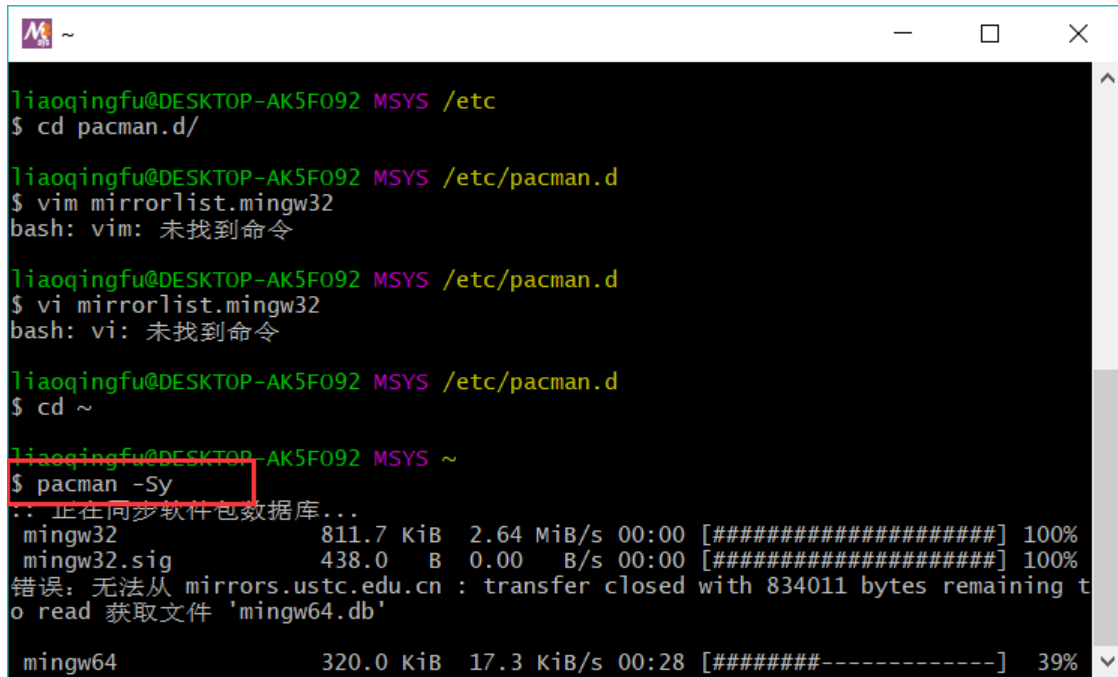
```
1 #进入msys2安装目录
2 D:
```



```
3 cd D:\msys64
4 #如果要打开msys2的mingw64窗口
5 msys2_shell.cmd -mingw64
6 #如果要打开msys2的msys窗口
7 #msys2_shell.cmd
```

然后在msys2的shell中执行：

```
1 pacman -Sy
```



```
liaoqingfu@DESKTOP-AK5F092 MSYS /etc
$ cd pacman.d/

liaoqingfu@DESKTOP-AK5F092 MSYS /etc/pacman.d
$ vim mirrorlist.mingw32
bash: vim: 未找到命令

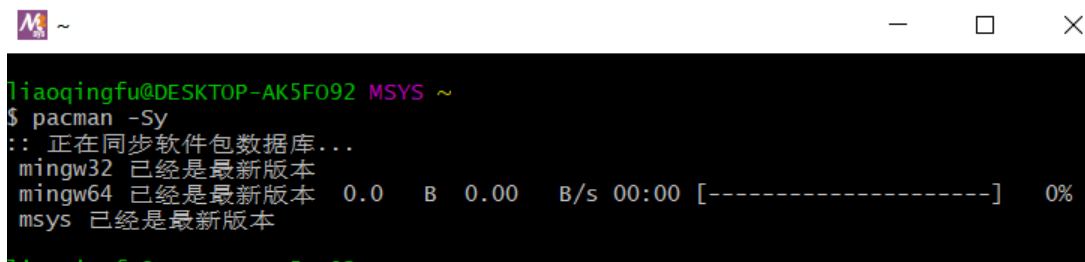
liaoqingfu@DESKTOP-AK5F092 MSYS /etc/pacman.d
$ vi mirrorlist.mingw32
bash: vi: 未找到命令

liaoqingfu@DESKTOP-AK5F092 MSYS /etc/pacman.d
$ cd ~

liaoqingfu@DESKTOP-AK5F092 MSYS ~
$ pacman -Sy
:: 正在同步软件包数据库...
 mingw32             811.7 KiB   2.64 MiB/s   00:00 [#####] 100%
 mingw32.sig         438.0 B     0.00 B/s   00:00 [#####] 100%
错误: 无法从 mirrors.ustc.edu.cn : transfer closed with 834011 bytes remaining to read 获取文件 'mingw64.db'

 mingw64             320.0 KiB   17.3 KiB/s   00:28 [#####-----] 39%
```

刷新软件包数据。



```
liaoqingfu@DESKTOP-AK5F092 MSYS ~
$ pacman -Sy
:: 正在同步软件包数据库...
 mingw32 已经是最新版本
 mingw64 已经是最新版本 0.0 B 0.00 B/s 00:00 [-----] 0%
 msys 已经是最新版本
```

## 4.3 安装编译环境

gcc编译器、git等

msys2 遇到两类开发环境：

1. MSYS2 自带的开发环境，安装的包叫 msys2-devel
2. MinGW-w64 的安装

这两者有什么区别呢？

一言以蔽之，前者编译出来的可执行文件，要依赖 MSYS2 提供的动态链接库，而后者不需要。下面详细说明一下：

(1) MSYS2 下的 gcc 编译环境，编译的可执行文件要依赖于 msys-2.0.dll，这个 DLL 提供了 Linux 下编程的提供的函数和接口，例如 fork 函数。

这个编译环境对于编译基于 Linux 下编写的软件，是非常适合的。例如编译 GNU 提供的各种工具。例如，你想编译最新版本的 GNU grep 工具，MSYS2 下的这个环境是非常适合的。

(2) 用 MinGW64 的编译环境，不再依赖于 msys-2.0.dll，如果源代码就是基于 windows 开发的，那使用 MinGW 的编译环境比较好，编译出来的可执行文件，不用再依赖 MSYS 提供的动态链接库。当然，前提是代码中不能使用 Linux 的东西，即 POSIX 的那套东西。

### 4.3.1 安装mingw-w64

**这里我们只用MinGW**

如果选择MinGW-w64编译则打开MSYS2 MinGW64，在shell窗口中输入：

```
1 pacman -S mingw-w64-x86_64-toolchain
```

然后默认全部安装即可（直接回车）。

而如果选择msys编译则打开MSYS2-MSYS2，在shell窗口中输入：

```
1 pacman -S msys2-devel
2 或者
3 pacman -S make gcc diffutils pkg-config
```

然后默认全部安装即可。

### 4.3.2 安装git

安装git：任一方式打开shell窗口输入：

```
1 pacman -S git
```

### 4.3.3 安装make等工具

```
1 pacman -S make
2 pacman -S automake
3 pacman -S autoconf
4 pacman -S perl
```

```
5 pacman -S libtool
6 pacman -S mingw-w64-i686-cmake
7 pacman -S pkg-config
8 如果需要编译出ffmpeg的话，还需要安装SDL
9 pacman -S mingw-w64-x86_64-SDL2
```

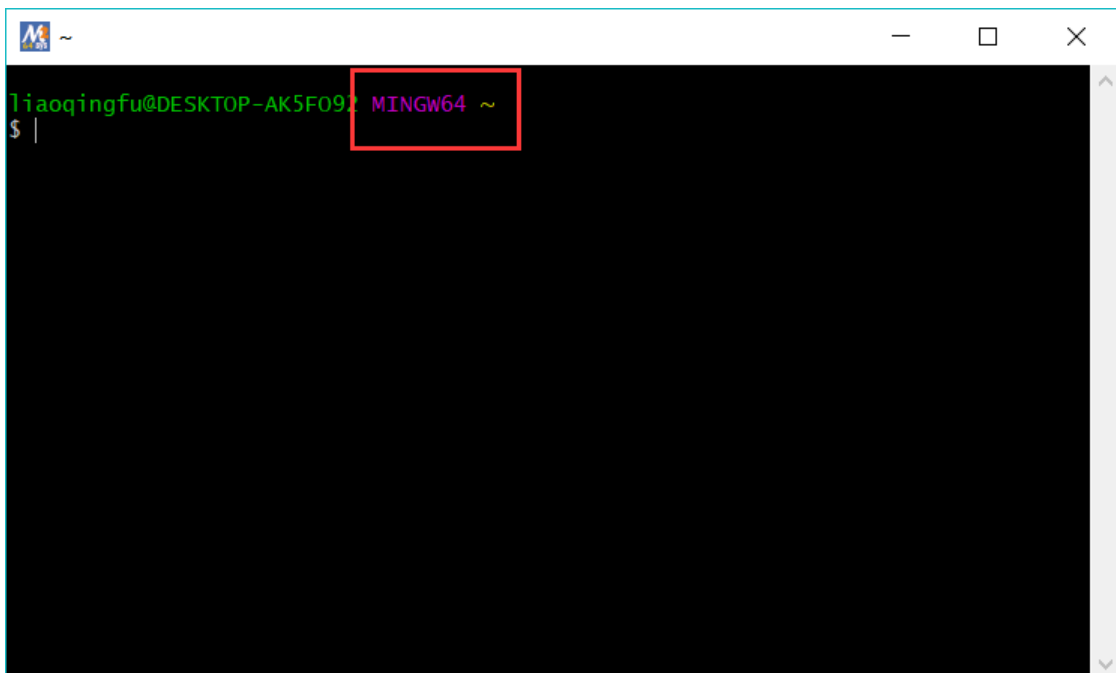
## 4.4 编译环境的其他准备工作

### 1.重命名link.exe

重命名 `msys64/usr/bin/link.exe` 为 `msys64/usr/bin/link.bak`，避免和MSVC 的link.exe抵触。

### 2.启动命令行窗口，在窗口中输入：

```
1 #进入msys2安装目录
2 D:
3 cd D:\msys64
4 #如果要打开msys2的mingw64窗口
5 msys2_shell.cmd -mingw64
6 #如果要打开msys2的msys窗口
7 #msys2_shell.cmd
```



### 3.下载和安装YASM

1. pacman -S yasm

### 4.检查编译环境工具

mingw64的shell窗口输入：

```
1 which cl link yasm cpp
```

看返回结果，没有no的结果一般就没问题。

```
liaoqingfu@DESKTOP-AK5F092 MINGW64 ~  
$ which cl link yasm cpp  
/c/Program Files (x86)/Microsoft Visual Studio 14.0/VC/BIN/amd64/cl  
/c/Program Files (x86)/Microsoft Visual Studio 14.0/VC/BIN/amd64/link  
/usr/bin/yasm  
/mingw64/bin/cpp
```

### 5.修改支持中文显示

窗口右键->Options->Text，然后locale选择：zh\_CN，Character set 选择 UTF-8。

### 6.安装nasm

编译当前最新x264时需要用到nasm。

```
1 pacman -S nasm
```

## 4.5 编译第三方库

### 4.5.1 下载和编译x264

#### 1.1。MinGW-w64版本

下载x264或者使用

将下载下的放置于/home/32687/ffmpeg，把第三方库编译的库文件放在 /home/32687/ffmpeg/build

```
1 git clone http://git.videolan.org/git/x264.git  
2 或者用码云的链接  
3 git clone https://gitee.com/mirrors_addons/x264.git
```

```
liaoqingfu@DESKTOP-AK5F092 MINGW64 ~  
$ git clone https://gitee.com/mirrors_addons/x264.git  
正克隆到 'x264'...  
remote: Enumerating objects: 31404, done.  
remote: Counting objects: 100% (31404/31404), done.  
remote: Compressing objects: 100% (5795/5795), done.  
remote: Total 31404 (delta 25647), reused 31272 (delta 25559), pack-reused 0  
接收对象中: 100% (31404/31404), 8.35 MiB | 658.00 KiB/s, 完成.  
处理 delta 中: 100% (25647/25647), 完成.
```

cd进入x264目录下：

```
1 ./configure --prefix=/home/32687/ffmpeg/build/libx264 --host=x86_64-w64-mingw32 --enable-static --extra-ldflags=-Wl,--output-def=libx264.def
2 make
3 make install
```

## 2.生成libx264.lib

上面编译出来的结果没有包含lib文件，需要自己手工生成。

configure时我们生成了libx264.def 此时就派上用场。

```
1 cp ./libx264.def /home/32687/ffmpeg/build/libx264/lib/
2 cd /home/32687/ffmpeg/build/libx264/lib
3 #若要生成64位lib文件则输入如下命令：
4 lib /machine:X64 /def:libx264.def
5 #若要生成32位lib文件则输入如下命令：
6 #lib /machine:i386 /def:libx264.def
```

即得到libx264.lib，然后将/home/32687/ffmpeg/build/libx264/bin/libx264-161.dll（具体名字和x264版本有关）改名或者复制一份为libx264.dll。

如果想在程序中直接使用x264的话，将include中的.h头文件、libx264.lib和libx264.dll复制到项目对应位置，并且在程序中添加头文件，然后就可以使用x264中的方法了。

加上

```
./configure --prefix=/home/32687/ffmpeg/build/libx264 --host=x86_64-w64-mingw32 --enable-shared --enable-static --extra-ldflags=-Wl,--output-def=libx264.def
```

**libx264官网下载**

<https://www.videolan.org/developers/x264.html>

## 4.5.2 下载和编译fdk-aac

下载fdk-aac

```
1 git clone --depth 1 https://gitee.com/mirrors/fdk-aac.git
2 cd fdk-aac
```

## 编译fdk-aac

```
1 ./autogen.sh
2 ./configure --prefix=/home/32687/ffmpeg/build/libfdk-aac --enable-
  static --enable-shared
3 make -j4
4 make install
```

### libfdk\_aac官网下载

<https://sourceforge.net/projects/opencore-amr/files/fdk-aac/>

## 4.5.3 下载编译mp3

下载

```
git clone --depth 1 https://gitee.com/hqiu/lame.git
```

编译

```
./configure --prefix=/home/32687/ffmpeg/build/libmp3lame --disable-shared --disable-
frontend --enable-static
make
make install
```

### libmp3lame官网下载（选择版本>= 3.98.3）

<https://sourceforge.net/projects/lame/files/lame/>

## 4.5.4 下载编译libvpx

```
git clone --depth 1 https://github.com/webmproject/libvpx.git
cd libvpx
./configure --prefix=/home/32687/ffmpeg/build/libvpx --disable-examples --disable-unit-
tests --enable-vp9-highbitdepth --as=yasm
make -j4
make install
```

## 4.6 下载和编译ffmpeg

### 1. 下载ffmpeg

```
1 git clone git://source.ffmpeg.org/ffmpeg.git
2 或者码云的链接
3 git clone https://gitee.com/mirrors/ffmpeg.git
4 cd ffmpeg
5 # 查看版本
6 git branch -a
7 # 选择4.2版本
8 git checkout remotes/origin/release/4.2
```

## git checkout remotes/origin/release/4.2

由于ffmpeg比较大，更好的选择官网下载ffmpeg。

## 2.编译ffmpeg

### 创建一个build.sh

将下载好的ffmpeg与x264放在一个目录下，本人是/home/32687/ffmpeg。

build\_ffmpeg.sh内容是：

```
1 ./configure \
2     --prefix=/home/32687/ffmpeg/build/ffmpeg-4.2 \
3     --arch=x86_64 \
4     --enable-shared \
5     --enable-gpl \
6     --enable-libfdk-aac \
7     --enable-nonfree \
8     --enable-libvpx \
9     --enable-libx264 \
10    --enable-libmp3lame \
11    --extra-cflags="-I/home/32687/ffmpeg/build/libfdk-aac/include" \
12    --extra-ldflags="-L/home/liaoqingfu/build/libfdk-aac/lib" \
13    --extra-cflags="-I/home/liaoqingfu/build/libvpx/include" \
14    --extra-ldflags="-L/home/liaoqingfu/build/libvpx/lib" \
15    --extra-cflags="-I/home/liaoqingfu/build/x264/include" \
16    --extra-ldflags="-L/home/liaoqingfu/build/x264/lib" \
17    --extra-cflags="-I/home/liaoqingfu/build/libmp3lame/include" \
18    --extra-ldflags="-L/home/liaoqingfu/build/libmp3lame/lib"
```

执行：

```
1 sh build_ffmpeg.sh
```

然后

```
make -j8
```

```
make install
```

## 4.7 MinGW与MSVC编译的区别

首先，MSVC是指微软的VC编译器，需要安装微软的VS软件，若是感觉软件比较庞大，可以安装visualcppbuildtools\_full，不过也很大哈！

然后，MinGW是指是Minimalist GNU on Windows的缩写。它是一个可自由使用和自由发布的Windows特定头文件和使用GNU工具集导入库的集合，允许你在GNU/Linux和Windows平台生成本地的Windows程序而不需要第三方C运行时库。感觉跨平台型更好呢

二：设置环境变量继承自Windows

找到msys2\_shell.cmd中的

```
rem set MSYS2_PATH_TYPE=inherit
```

去掉rem，取消这一句的注释，使MSYS2的环境变量继承自系统

## 5 QT使用MSVC2015 64bit调用我们编译的ffmpeg库

xxx.pro范例

```
1 TEMPLATE = app
2 CONFIG += console
3 CONFIG -= app_bundle
4 CONFIG -= qt
5
6 SOURCES += \
7     main.c
8
9 INCLUDEPATH += \
10     $$PWD/ffmpeg-4.2/include
```



```
11 LIBS += $$PWD/ffmpeg-4.2/bin/avformat.lib \
12     $$PWD/ffmpeg-4.2/bin/avcodec.lib \
13     $$PWD/ffmpeg-4.2/bin/avdevice.lib \
14     $$PWD/ffmpeg-4.2/bin/avfilter.lib \
15     $$PWD/ffmpeg-4.2/bin/avutil.lib \
16     $$PWD/ffmpeg-4.2/bin/swresample.lib \
17     $$PWD/ffmpeg-4.2/bin/swscale.lib
```

## 测试文件

```
1 #include <stdio.h>
2
3 // 包含ffmpeg头文件
4 #include "libavutil/avutil.h"
5
6 int main()
7 {
8     printf("Hello FFmpeg, version is %s\n", av_version_info());
9
10    return 0;
11 }
```