ffmpeg编译

第一步 安装NDK

下载最新NDK

> *https://developer.android.google.cn/ndk/downloads/index.html*

配置NDK环境

- 解压下载的NDK

```
unzip android-ndk-r20-linux-x86_64.zip -d ndk
```

- 配置环境变量

1. 修改~/.bashrc

```
vim ~/.bashrc
# 在文件末尾添加
export NDKROOT=/home/fanzhang/ndk/android-ndk-r20
exprot PATH=$NDKROOT:$PATH
# 保存退出，更新一下环境变量
source ~/.bashrc
```

2. 修改 "/etc/"下面的profile文件

```
vim /etc/profile
# 在文件末尾添加
export NDKROOT=/home/fanzhang/ndk/android-ndk-r20
exprot PATH=$NDKROOT:$PATH
# 保存退出，更新一下环境变量
source /etc/profile
```

3. 可以在shell中输入`ndk-build`命令来检查你的安装是否成功，如果不是显示“ndk-build not found”, 则说明你的ndk安装成功

ffmpeg

> *git clone git://source.ffmpeg.org/ffmpeg.git ffmpeg*

编译ffmpeg

- 修改configure 首先需要对源代码中的configure文件进行修改，由于编译出来的动态库文件名的版本号在.so之后(例如“libavcodec.so.5.100.1”),而Android平台不能识别这样的文件名，所以需要修改这种文件名。找到FFMpeg文件夹下面的configure文件，做如下修改

```
SLIBNAME_WITH_MAJOR='$(SLIBNAME).$(LIBMAJOR)'
LIB_INSTALL_EXTRA_CMD='$$(RANLIB) "$(LIBDIR)/$(LIBNAME)"'
SLIB_INSTALL_NAME='$(SLIBNAME_WITH_VERSION)'
SLIB_INSTALL_LINKS='$(SLIBNAME_WITH_MAJOR) $(SLIBNAME)'
将其修改成：
SLIBNAME_WITH_MAJOR='$(SLIBPREF)$(FULLNAME)-$(LIBMAJOR)$(S
LIBSUF)'
LIB_INSTALL_EXTRA_CMD='$$(RANLIB) "$(LIBDIR)/$(LIBNAME)"'
SLIB_INSTALL_NAME='$(SLIBNAME_WITH_MAJOR)'
SLIB_INSTALL_LINKS='$(SLIBNAME)'
```

- 编写 编译脚本build_bash.sh

```
#!/bin/bash
NDK=/home/fanzhang/ffmpegwork/ndk/android-ndk-r20
TOOLCHAIN=$NDK/toolchains/llvm/prebuilt/linux-x86_64/
API=29

function build_android
{
echo "Compiling FFmpeg for $CPU"
./configure \
    --prefix=$PREFIX \
    --disable-neon \
    --disable-hwaccels \
    --disable-gpl \
    --disable-postproc \
    --enable-shared \
    --enable-jni \
    --disable-mediacodec \
    --disable-decoder=h264_mediacodec \
    --disable-static \
    --disable-doc \
    --disable-ffmpeg \
    --disable-ffplay \
    --disable-ffprobe \
    --disable-avdevice \
    --disable-doc \
    --disable-symver \
    --cross-prefix=$CROSS_PREFIX \
    --target-os=android \
    --arch=$ARCH \
    --cpu=$CPU \
    --cc=$CC
    --cxx=$CXX
```

```
    --enable-cross-compile \
    --sysroot=$SYSROOT \
    --extra-cflags="-Os -fpic $OPTIMIZE_CFLAGS" \
    --extra-ldflags="$ADDI_LDFLAGS" \
    $ADDITIONAL_CONFIGURE_FLAG
make clean
make
make install
echo "The Compilation of FFmpeg for $CPU is completed"
}

#armv8-a
ARCH=arm64
CPU=armv8-a
CC=$TOOLCHAIN/bin/aarch64-linux-android$API-clang
CXX=$TOOLCHAIN/bin/aarch64-linux-android$API-clang++
SYSROOT=$NDK/toolchains/llvm/prebuilt/linux-x86_64/sysroot
CROSS_PREFIX=$TOOLCHAIN/bin/aarch64-linux-android-
PREFIX=$(pwd)/android/$CPU
OPTIMIZE_CFLAGS="-march=$CPU"
build_android

#armv7-a
ARCH=arm
CPU=armv7-a
CC=$TOOLCHAIN/bin/armv7a-linux-androideabi$API-clang
CXX=$TOOLCHAIN/bin/armv7a-linux-androideabi$API-clang++
SYSROOT=$NDK/toolchains/llvm/prebuilt/linux-x86_64/sysroot
CROSS_PREFIX=$TOOLCHAIN/bin/arm-linux-androideabi-
PREFIX=$(pwd)/android/$CPU
OPTIMIZE_CFLAGS="-mfloat-abi=softfp -mfpu=vfp -marm -
march=$CPU "
build_android

#x86
ARCH=x86
CPU=x86
CC=$TOOLCHAIN/bin/i686-linux-android$API-clang
CXX=$TOOLCHAIN/bin/i686-linux-android$API-clang++
SYSROOT=$NDK/toolchains/llvm/prebuilt/linux-x86_64/sysroot
CROSS_PREFIX=$TOOLCHAIN/bin/i686-linux-android-
PREFIX=$(pwd)/android/$CPU
OPTIMIZE_CFLAGS="-march=i686 -mtune=intel -mssse3 -
mfpmath=sse -m32"
build_android

#x86_64
ARCH=x86_64
CPU=x86-64
CC=$TOOLCHAIN/bin/x86_64-linux-android$API-clang
CXX=$TOOLCHAIN/bin/x86_64-linux-android$API-clang++
SYSROOT=$NDK/toolchains/llvm/prebuilt/linux-x86_64/sysroot
```

```
CROSS_PREFIX=$TOOLCHAIN/bin/x86_64-linux-android-
PREFIX=$(pwd)/android/$CPU
OPTIMIZE_CFLAGS="-march=$CPU -msse4.2 -mpopcnt -m64 -
mtune=intel"
build_android
```

通用的编译第三方so库的办法

1. ubuntu 安装ndk 配置ndk环境
2. 把第三方库的源码拉下来 git
3. 官网看下有没有提供编译脚本build.sh readme.txt configure
4. 编译so库出来了