**ijkplayer下载与编译**

一、脚本、代码下载与编译

1、脚本与ijkplayer代码下载

git clone git@gitlab.ximalaya.com:jsyan/ijk.git

2、ijkplayer代码分支切换

cd ijk/

Android分支：remotes/origin/ijk-for-xm

Ios分支：remotes/origin/ijk-for-xm-ios

3、ffmpeg代码下载

./init-android.sh

下载ffmpeg、libyuv代码，并配置ffmpeg编译模块选项module.sh

4、编译ffmpeg

cd android/contrib

./compile-ffmpeg all

5、编译ijkplayer

cd ..

./compile-ijk all

6、demo apk编译

./gradlew build --stacktrace

二、脚本分析

1、init-android.sh

sh tools/pull-repo-base.sh git@gitlab.ximalaya.com:jsyan/ijk-ffmpeg-xm-fork.git extra/ffmpeg

1.1、pull-repo-base.sh

git clone git@gitlab.ximalaya.com:jsyan/ijk-ffmpeg-xm-fork.git extra/ffmpeg

或

git fetch --all --tags

下载ffmpeg代码到目录extra/ffmpeg

1.2、pull\_fork

pull\_fork "armv5"

pull\_fork "armv7a"

pull\_fork "arm64"

pull\_fork "x86"

pull\_fork "x86\_64"

pull\_fork：

sh tools/pull-repo-ref.sh [git@gitlab.ximalaya.com:jsyan/ijk-ffmpeg-xm-fork.git](mailto:git@gitlab.ximalaya.com:jsyan/ijk-ffmpeg-xm-fork.git) android/contrib/ffmpeg-$1 extra/ffmpeg

1.3、pull-repo-ref.sh

git clone --reference extra/ffmpeg git@gitlab.ximalaya.com:jsyan/ijk-ffmpeg-xm-fork.git android/contrib/ffmpeg-$1

ffmpeg代码拷贝到 android/contrib/ffmpeg-\*目录下，比如：android/contrib/ffmpeg-armv7a

cd android/contrib/ffmpeg-\*

git checkout ff3.1--ijk0.6.2--20160926--001 -B ijkplayer创建基于ff3.1 tag的分支ijkplayer。

这里的代码已经和extra/ffmpeg下不同。

1.4、init-config.sh

配置ffmpeg的编译选项，配置文件module-lite.sh

cp config/module-lite.sh config/module.sh

1.5、init-android-libyuv.sh

下载libyuv代码：

sh tools/pull-repo-base.sh [git@gitlab.ximalaya.com:jsyan/ijk-libyuv-xm-fork.git](mailto:git@gitlab.ximalaya.com:jsyan/ijk-libyuv-xm-fork.git) extra/libyuv

sh tools/pull-repo-ref.sh [git@gitlab.ximalaya.com:jsyan/ijk-libyuv-xm-fork.git](mailto:git@gitlab.ximalaya.com:jsyan/ijk-libyuv-xm-fork.git) ijkmedia/ijkyuv extra/libyuv

cd ijkmedia/ijkyuv

git checkout ijk-r0.2.1-dev

下载libyuv代码到extra/libyuv，

下载libyuv代码到ijkmedia/ijkyuv，tag切换到ijk-r0.2.1-dev

2、ffmpeg编译

cd android/contrib

./compile-ffmpeg.sh all

2.1、compile-ffmpeg.sh

执行tools/do-compile-ffmpeg.sh armv5 armv7a arm64 x86 x86\_64分别编译出这五种架构下的动态库。

2.2、tools/do-compile-ffmpeg.sh

简要流程：

a、arch选择，配置编译环境选择编译工具链等，例如armv7a选择armeabi-v7a、arm-linux-androideabi

b、编译输出路径android/contrib/build/ffmpeg-armv7a/output

c、NDK配置工具链 --platform --toolchain，得到

${FF\_CROSS\_PREFIX}-gcc

${FF\_CROSS\_PREFIX}-ld

${FF\_CROSS\_PREFIX}-ar

d、cd $FF\_SOURCE 进入ffmpeg源码位置，比如android/contrib/ffmpeg-armv7a下。

运行./configure，得到config.h文件。

e、使用GCC编译并链接后生成动态库libijkffmpeg-$FF\_LIBNAME.so

3、编译ijkplayer

cd ijk/android

./compile-ijk.sh all

3.1、compile-ijk.sh

进入源代码目录ijkplayer/ijkplayer-$PARAM\_TARGET/src/main/jni

利用Android.mk编译，大概有三个部分：

1、ndk prof

2、ffmpeg动态库预加载

3、ijkmedia目录，软链接到ijk/ijkmedia

3.2、ijkmedia目录编译

有4个部分：

1. ijkj4a目录，编译成静态库libijkj4a.a，功能是抽象封装jni中回调java类及类方法

2、libyuv目录，编译成libyuv\_static.a

3、ijksdl目录，依赖ijkffmpeg、ijkj4a、libyuv动态库，编译成动态库libijksdl-xxx.so

4、ijkplayer目录，依赖ijkffmpeg、ijksdl动态库，编译成动态库libijkplayer-xxx.so

编译完成，输出路径：ijk/android/ijkplayer/ijkplayer-\*/src/main/libs/\*

libijkplayer-armeabi-v7a.so、libijksdl-armeabi-v7a.so。

4、demo APK编译

ijk/android/ijkplayer目录使用./gradlew build

输出：XMPlayer-all32-debug.apk