**QMediaPlayer**

Downloads prebuild from here = <https://ffmpeg.zeranoe.com/builds/>

https://wiki.videolan.org/AndroidCompile/

<https://developer.android.com/guide/topics/media/mediaplayer.html>

<https://developer.android.com/guide/topics/media/media-formats.html>

http://asteria-00:8080/job/MCS\_CloudInterface/ user : tester password : asteria2015

ffmpeg -f dshow -i video="screen-capture-recorder":audio="Stereo Mix (IDT High Definition" -vcodec libx264 -preset ultrafast -tune zerolatency -r 10 -async 1 -acodec libmp3lame -ab 24k -ar 22050 -bsf:v h264\_mp4toannexb -maxrate 750k -bufsize 3000k -f mpegts udp://@224.10.10.10:15004/

working:

How to Compile FFMPEG : https://trac.ffmpeg.org/wiki/CompilationGuide/MSVC

<http://dranger.com/ffmpeg/tutorial01.html>

<http://stackoverflow.com/questions/4725773/ffmpeg-on-android>

<http://www.qtav.org/install.html>

<http://dranger.com/ffmpeg/>

<http://www.mediaentertainmentinfo.com/2014/01/1-technical-series-how-to-compile-ffmpeg-under-cygwin.html/>

<https://forum.qt.io/topic/42213/module-problem-using-qtav>

Error : :-1: error: LNK1181: cannot open input file '..\.obj\QtAV\_win\_x86\CopyFrame\_SSE2.obj'

Copy paste file from build-QtAV-Desktop-Release\src to build-QtAV-Desktop-Release\.obj\QtAV\_win\_x86

Error: common. Not found rename “common.lib” to “common.”

<https://mr-johal.com/album/52991/english-top-7-songs-july-2016.html>

ANDROID Lib <https://sourceforge.net/projects/ffmpeg4android/files/20140318/>

http://stackoverflow.com/questions/16033900/what-is-the-ip-address-of-android-emulator

<http://stackoverflow.com/questions/15350555/set-ip-address-for-android-emulator>

http://www.biemmeitalia.net/blog/android-network-configuration/

Cd C:/Program Files (x86)/Android/android-sdk/platform-tools/adb.exe

adb.exe -s emulator-5554 shell

# netcfg

# tcpdump -I eth0 host 10.0.2.2 (machine port 8080)

# netstat -s –udp

# top -m 5

<http://stackoverflow.com/questions/5064304/how-can-i-forward-my-localhost-ip-address-to-an-android-emulator?rq=1>

Telnet localhost <Port of you'r emulator> (5554 for me)

redir add udp:5000:6000

ANDROID EMULATOR

Host machine can be reached using IP address 10.0.2.2 from the emulator.

These are the [IP addresses as reached from the emulator](https://developer.android.com/studio/run/emulator-commandline.html#emulatornetworking):

* 10.0.2.1, Router/gateway address.
* 10.0.2.2, Special alias to your **host** loopback interface (i.e., 127.0.0.1 on your development machine)
* 10.0.2.3, First DNS server
* 10.0.2.4 / 10.0.2.5 / 10.0.2.6, Optional second, third and fourth DNS server (if any)
* 10.0.2.15, The **emulated** device's own **network/ethernet interface**
* 127.0.0.1, The **emulated** device's own **loopback interface**

That said, we have:

* Common mistake 1: accessing 127.0.0.1 from the emulator trying to reach your host machine. Use 10.0.2.2, as I said.
* Common mistake 2: Trying to access an emulator service on HostComputerIP:appServicePort. It won't work since your host computer itself (Windows, Linux, OS etc.) is not running a service in that port. You need to redirect a port on the emulator **console** to a port on an emulated **Android instance** itself (see 2 below).

Common networking needs:

**1- Emulator app as client and local computer as server**

Because the emulator is NAT'd, I believe you can connect to any computer on your local network directly. I mean, since the virtual router has access to both networks, it should be able to handle outgoing (i.e., emulator->real lan) connections just fine.

Example: on my network (192.168.0.x), I can connect from the emulator to my real router (192.168.0.254) just pointing the emulator web browser to http://192.168.0.254:port. I use different services on it (hail to Tomato!), and I can access all of them on each port. No need to handle port forwarding, as expected.

**By the looks of your code, I believe you need:**

// I assume 192.168.0.114 is your server, which is

// located on your local network, running a server application

// on port 9999.

cSocket = new Socket("192.168.0.114",9999);

**2- Local computer as client and emulator app as server**

Now that's a different story. You need to setup [port redirections](https://developer.android.com/studio/run/emulator-commandline.html#redirection) on the virtual router. The easiest way is:

Telnet into the "management" system (this is not the emulator), from your **host** (your computer, console on linux or command prompt on Windows):

telnet localhost 5554

After that, use:

adb forward tcp:localPort tcp:emulatorPort

After this, you will be able to have a service on emulatorPort and you will be able to connect to it from computers in the local network by accessing hostComputerIP:localPort.

This is the way people (including me) use, for example, SSHDroid inside an emulator.

<https://wiki.qt.io/How-to-debug-Qt-applications-on-Android-device>

<https://google.github.io/ExoPlayer/guide.html>

http://doc.qt.io/qt-5/qquickimageprovider.html