
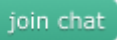

PSA - WebRTC builds have moved to using GN instead of GYP. Android build script is adapted, but iOS script is still break. Feel free to fork and update them.

WebRTC Build Scripts

  A set of build scripts useful for building WebRTC libraries for Android and iOS.

Bugs: Please submit the revision number that you are using. There are frequent updates to this project so please watch the changelist for bug fixes.

Android ARMv7, ARMv8, x86, x86_64 Builds – Guide here

The following instructions are for building the native WebRTC libraries for Android.

Getting Started

On Linux The scripts can probably work on most distros, although we only have experience with Ubuntu 12.04 and 14.04 on 64 bit machines.

This is only required once.

```
1
2 # Source all the routines
3 source android/build.sh
4
5 # Install any dependencies needed
6 install_dependencies
7
8 # Pull WebRTC
9 get_webrtc
```

On Mac or Windows If you don't have a Ubuntu machine available, or you are too lazy to setup a virtual machine manually, you can build WebRTC for Android on your Mac or Windows PC through our Vagrant script.

First of all, you need to download and install Vagrant. After that, from the `/android` directory, you need to execute the following in you shell:

```
1
2 # If you need to use private SSH keys from your host computer
3 # Execute this line of code to ensure your private key is added to your
  identity
```

```
4  ssh-add -L
5
6  # If there are no identities, add them by:
7  ssh-add ~/.ssh/id_rsa
8
9  # Boot up and provision the Vagrant box
10 vagrant up
11
12 # SSH into the Vagrant box
13 vagrant ssh
14
15 # Installs the required dependencies on the machine
16 install_dependencies
```

On Windows machines you may face issues with long path names on the VM that aren't handled correctly. A work around is to copy the script to another directory (not the one shared between the VM and Windows host), and build there:

```
1
2 mkdir mybuild
3 cd mybuild
4 cp /vagrant/build.sh .
5 source ./build.sh
6 get_webrtc
7 build_apprtc
```

Building the libraries Then you can build the Android example

```
1 # Pull WebRTC
2 get_webrtc
3
4 # Build apprtc
5 build_apprtc
6
7 # Build in debug mode
8 export WEBRTC_DEBUG=true
9 build_apprtc
```

You can build for armv7, armv8, x86, x86_64 platform

```
1 export WEBRTC_ARCH=armv7 #or armv8, x86, or x86_64
2 prepare_gyp_defines &&
3 execute_build
```

You can build a particular revision

```
1 # Pull WebRTC
2 get_webrtc 6783
3
```

```
4 # Build apprtc
5 build_apprtc
```

When the scripts are done you can find the .jar and .so file in \$WEBRTC_ROOT under “libjingle_peerconnection_builds”.

iOS (armv7, arm64, i386) and Mac (X86_64) – Guide here

These steps must be run on Mac OSX

Source the ios build scripts or open the Xcode project

```
1 source ios/build.sh
```

Specify if you want to build for Debug/Profile/Release by setting either WEBRTC_DEBUG, WEBRTC_PROFILE, WEBRTC_RELEASE as an environment variable in your bash or xcode scheme run settings.

```
1 WEBRTC_DEBUG=true
2 WEBRTC_PROFILE=true
3 #or
4 WEBRTC_RELEASE=true
```

Building the libraries Then you can build the iOS example

```
1 # We use the term webrtc dance a lot to build
2 dance
3
4 # Or in two steps
5 get_webrtc
6 # Make changes then build WebRTC
7 build_webrtc
```

Mac example

```
1 # Get WebRTC
2 get_webrtc
3 # Make changes then build WebRTC
4 build_webrtc_mac
```

Check which revision you are using at ./webrtc-build-scripts/ios/webrtc/libWebRTC-LATEST-Universal-Debug.a.version.txt

Open the xcode project, and execute the AppRTC Demo on any iOS 7 device or simulator

```
1 open ./webrtc-build-scripts/ios/WebRTC.xcodeproj
```

You can also build a particular revision

```
1 #Pull WebRTC
2 update2Revision 6783
```

Make changes then,

```
1 #Build WebRTC
2 build_webrtc
```

Make sure you label your new binaries that are generated in

```
1 ./webrtc-build-scripts/ios/webrtc/libjingle_peerconnection_builds
```

Cocoapods!!

platform ios | osx

platform ios | osx

platform ios | osx

Usage

To run the example AppRTC Demo project, clone the repo, and run `pod install` from the Example directory first.

Requirements

A fast internet connection.... for your own sanity

Installation

libjingle_peerconnection starting from revision 6931 is available through CocoaPods. To install it, simply add the following line to your Podfile:

```
1 pod "libjingle_peerconnection"
```

iOS ARM64 builds are available as of 7810.0.0

mac x86_64 builds are available as of 7759.0.0

###Linux x86, x86_64 Builds

The following instructions are for building the native WebRTC libraries for Linux.

Getting Started

On Linux The scripts can probably work on most distros, although we only have experience with Ubuntu 12.04, 14.04, 16.04 and 16.10 on 64 bit machines.

This is only required once.

```
1
2 # Source all the routines
3 source linux/build.sh
4
5 # Install any dependencies needed
6 install_dependencies
7
8 # Pull WebRTC
9 get_webrtc
10
11 # Build apprtc
12 build_apprtc
```

You can build for arm-linaro-gnueabi, x86, x86_64 platform

```
1 export WEBRTC_ARCH=x86 #, arm-linaro-gnueabi or x86_64
2 prepare_gyp_defines &&
3 execute_build
```

You can build a particular revision

```
1 # Pull WebRTC
2 get_webrtc 6783
3
4 # Build apprtc
5 build_apprtc
```

Versioning

The versioning can be explained as follows:

6931.2.0

6931 reflects the SVN revision from the WebRTC root Google Code Project

2 reflects a Release Build (0 for Debug, 1 for Profile)

Profile builds are no longer built by default

The minor 0 reflects any changes I might need to make to the sample xcode project itself to work (incremented normally)