1. 下载
2. 搭建环境

Chromium的源码管理采用google 的 depot\_tools工具，基本环境为：

Ubuntu 14及其以上的64位系统、100G左右的空间、16G的内存、VPN（速度必须快）

1. 通过apt-get install git的时候，可能会出现找不到local package的错误，此时，需要通过命令 sudo apt-get update去更新source-list表
2. 安装git工具

sudo apt-get install git

1. 配置git的http代理

git config --global http.proxy 10.10.121.148:1080

git config --global https.proxy 192.168.211.1:1080

1. 安装depot\_tools

git clone

<https://chromium.googlesource.com/chromium/tools/depot_tools.git>

1. 配置depot\_tools

export PATH=$PATH:/path/to/depot\_tools

1. 获取chromium源码

mkdir ~/chromium && cd ~/chromium

fetch --nohooks --no-history android

如果中途被打断，可用命令如下继续下载

gclient sync –nohooks

下载完成会有completed提示

注：必须加—no-history，否则源码的大概有30多G，下载时间如果是下载速度为1M/s的网络，需要下载三天至少，因为中间可能应为一些不知道的原因导致下载中断

1. 编译
2. 安装JDK

sudo add-apt-repository ppa:webupd8team/java

sudo apt-get update

安装JDK8

sudo apt-get install oracle-java8-installer

设置系统默认JDK8

sudo update-java-alternatives -s java-8-oracle

测试

java –version

javac -version

注：必须安装JDK，否则在执行build/install-build-deps-android.sh的时候，会报找不到JDK的错误。

1. 安装依赖环境

build/install-build-deps-android.sh

这个过程中会弹出一些选择框，按Tab+Enter键进行OK确认

安装完成会有提示：

install-build-deps-android.sh complete.

注：系统必须是Ubuntu 14及其以上，否则执行会中断

1. 配置GYP环境

echo "{ 'GYP\_DEFINES': 'OS=android target\_arch=arm', }"

> chromium.gyp\_env

gclient runhooks

注1：在运行gclient runhooks的时候，会出现执行/usr/bin/python src/build/android/play\_services/update.py download的时候，socket timeout的错误，此时需要设置python的socket5代理，具体方法为：

下载PySocks，路径：

[https://superb-sea2.dl.sourceforge.net/project/socksipy/socksipy/ SocksiPy%201.00/SocksiPy.zip](https://superb-sea2.dl.sourceforge.net/project/socksipy/socksipy/%20SocksiPy%201.00/SocksiPy.zip)

解压，拷贝socks.py这个文件至depot\_tools目录，并且在gsutils.py中，添加：

import socks

import socket

socks.setdefaultproxy(socks.PROXY\_TYPE\_SOCKS5,"10.10.121.148",1080)

socket.socket =socks.socksocke

然后重启虚拟机，即可成功执行。

注2：执行的过程大概需要半个小时

注3：执行完成的标志是：running '/usr/bin/python src/third\_party/gvr-android-sdk/test-apks/update.py' in '/home/zdx/chromium'

1. 生成ninja编译配置

gn gen --args='target\_os="android"' out/Default

注1：如果gn help能出现提示，表示gclient runhooks成功了，此时也可以执行gn命令

注2：在执行此命令之前只有out目录，如果执行成功，则会生成Default目录，并且Default目录下面包含文件如下：



1. 编译

编译命令：

gn gen out/Release --args='target\_os="android"'

ninja -C out/Release system\_webview\_apk

注1：编译依赖于服务器性能，如果用虚拟机编译，设置双核四线程，内存3GB，但是在编译的过程中，仍然出现了错误提示：internal compiler error: Killed (program cc1plus)，从网上的资料中可以发现，这是因为服务器内存不足造成的。因此我们需要要么扩展虚拟机内存，要么采用内存交换的方式。

内存交换：

创建分区文件，大小4GB

dd if=/dev/zero of=/swapfile bs=1k count=4096000

生成swap文件系统

mkswap /swapfile

激活swap文件

swapon /swapfile

修改 /etc/fstab 文件，新增如下内容

/swapfile swap swap defaults 0 0

注2：编译采用酷睿i3双核+4GB内存+100GB硬盘上面的虚拟机，时间需要3天左右。

1. 编译生成的SystemWebView.apk可以正常在Android6.0之后的机顶盒上面运行