

浏览器 FAQ

文件标识: RK-PC-YF-465

发布版本: V1.2.0

日期: 2023-02-16

文件密级: 绝密 秘密 内部资料 公开

免责声明

本文档按“现状”提供，瑞芯微电子股份有限公司（“本公司”，下同）不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因，本文档将可能在未经任何通知的情况下，不定期进行更新或修改。

商标声明

“Rockchip”、“瑞芯微”、“瑞芯”均为本公司的注册商标，归本公司所有。

本文档可能提及的其他所有注册商标或商标，由其各自拥有者所有。

版权所有 © 2023 瑞芯微电子股份有限公司

超越合理使用范畴，非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园A区18号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

前言

概述

产品版本

芯片名称	内核版本
全系列	通用

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

修订记录

版本号	作者	修改日期	修改说明
V1.0.0	陈谋春	2017-05-17	
V1.1.0	陈谋春	2020-09-15	增加百度问题说明
V1.1.1	陈谋春	2022-05-26	更新文件标识
V1.2.0	陈谋春	2023-02-16	增加浏览器闪退问题说明

目录

浏览器 FAQ

1. Webview & Browser & Chrome
2. HTML5
3. Webview FAQ
 - 3.1 如何升级 Webview
 - 3.2 视频无法播放
 - 3.2.1 手势限制
 - 3.2.2 安全限制
 - 3.2.3 其他问题
 - 3.3 视频卡顿
 - 3.4 视频无法循环或自动播放
 - 3.5 动画或游戏卡顿
 - 3.6 黑屏、白屏和闪烁
 - 3.7 如何修改 UserAgent
 - 3.8 如何实现网址过滤
 - 3.9 无法打开下载文件
 - 3.10 是否支持 Adobe Flash
 - 3.11 崩溃和 ANR 如何处理
 - 3.12 视频有声无影
 - 3.13 浏览器上百度首页，会报错：net::err_unknown_url_scheme
 - 3.14 所有调用webview的浏览器都闪退，logcat提示java.lang.ClassNotFoundException: Invalid name:
4. How to build Webview
5. How to debug
6. 重装webview

1. Webview & Browser & Chrome

在本文开始前，有必要明确一下这三者的差别，Webview 是 Android 框架层的核心组件，所有应用都可以通过内嵌 Webview 的方式很方便的集成 Web 功能，而不需要自己去移植庞大复杂的 Web Engine；而 Browser 则是 Android 提供的一个全功能网页浏览器，其本质也是通过 Webview 来实现的；最后一个 Chrome 则是基于 Chromium 开源工程的，和桌面上最流行的 Chrome 浏览器是同一份代码编译的。

2. HTML5

HTML5 是 W3C 最新的 Web 标准，用来取代之前的 HTML、XHTML 以及 HTML DOM，增加了很多新的特性：

- 用于绘画的 canvas 元素
- 用于媒介回放的 video 和 audio 元素
- 对本地离线存储的更好的支持
- 新的特殊内容元素，比如 article、footer、header、nav、section
- 新的表单控件，比如 calendar、date、time、email、url、search

很多人把 HTML5 和音视频、游戏划等号，这其实是不对，HTML5 包含的新特性非常多，我上面列的也并不完整，有兴趣的可以看完整的[HTML5 规范](#)。

3. Webview FAQ

3.1 如何升级 Webview

Web Engine 本身非常庞大，外围依赖模块又多，并且又是完全开放的，所以随着系统组件或网页内容的更新，难免会有一些 BUG 和兼容性问题，目前最新的 Webview 已经是基于 Chromium 主线分支，版本发布和 Chrome 一样，也是每个月一个稳定版本。所以对于 Browser，还有其他基于 Webview 实现的应用，如果有碰到未知问题，都可以尝试升级 Webview 看能否解决。Android 5.1 开始大致的升级步骤如下（老版本的 Android 和最新版本的 Webview 不兼容）：

- Step1: 选择一个稳定版本

Android 的 Webview 目前有很多发行版，具体如下(只讨论稳定版本):

Name	PackageName	获取方式	自动更新 1	稳定性

Name	PackageName	获取方式	自动更新 1	稳定性
Android WebView	com.android.webview	Android 自带	否	最高
Chrome Stable ²	com.android.chrome	Chrome 自带	可	高
Google WebView ³	com.google.android.webview	随 GMS 包发布	可	高
Custom Webview	com.android.webview	自编译	否	中

从上表可以看出，*Google WebView & Chrome Stable*都可以通过 Google Play 来升级，通过 GMS 认证的机器默认就是用的这两种 Webview，要升级也最容易，可以直接升级 GMS 包，也可以单独替换 Webview 的 APK。

- Step2: 修改系统 Webview 的包名(可选)

Android 5.1 开始，Webview 具体实现从框架层剥离出来，通过一个包名来控制加载真正的 Webview 实现，默认的包名是*com.android.webview*，如果要切换到不同的 Webview 实现，就要先改掉系统默认的包名，具体修改办法如下（如果只升级版本，不切换发行版可跳过这一步）：

1. For android 6.0 & before

老版本的 Android 配置文件是/*path_to_android/frameworks/base/core/res/values/config.xml*，其中相关配置如下：

```
<string name="config_webViewPackageName"
translatable="false">com.android.webview</string>
```

其中 *com.android.webview* 可以改成你要切到的发行版包名，例如 *com.google.android.webview*。

2. For android 7.0 & after

新版本的 Android 配置文件

是/*path_to_android/frameworks/base/core/res/xml/config_webview_packages.xml*，改成如下配置：

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Copyright 2015 The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
See the License for the specific language governing permissions and
limitations under the License.
--&gt;

&lt;webviewproviders&gt;
    &lt;!-- The default WebView implementation --&gt;</pre>

```

```
<webviewprovider description="Android WebView"  
packageName="com.android.webview" availableByDefault="true"></webviewprovider>  
    <webviewprovider description="Chrome Stable"  
packageName="com.android.chrome" availableByDefault="true" />  
    <webviewprovider description="Google WebView"  
packageName="com.google.android.webview" availableByDefault="true"  
isFallback="true" />  
</webviewproviders>
```

系统在开机过程中会自动根据这个配置文件中的顺序来搜索设备中已安装并启用的包信息，找到以后直接返回，例如上面配置中的三个发行版如果都安装并启用了，则默认的包名是`com.android.webview`。我这里不建议修改这三个发行版的顺序，因为本身已经是按稳定性排序过的了。

- Step3: 手动安装并测试

```
# Uninstall any webview updates  
adb uninstall com.google.android.webview # 失败也没关系  
adb uninstall com.android.webview # 失败也没关系  
# On Android 8.0 and up:  
adb disable-verity; adb reboot  
  
# Remove webview from system partition  
adb root  
adb remount  
adb shell stop  
adb shell rm -rf /system/app/webview /system/app/WebViewGoogle  
/system/app/WebViewStub  
adb shell start  
Install the built apk.  
adb install -r -d out/Release/apks/SystemWebView.apk
```

安装完成后就可以验证问题是否得到解决，以及是否带来新的问题。

- Step4: 集成到发布固件

验证 ok 以后就需要集成到固件中发布，具体如下：

1. For android 5.1

```
# webview_xxx.apk即你要替换的WebView APK  
cp webview_xxx.apk webview.zip  
mv webview_xxx.apk webview.apk  
  
# 解压获取libwebviewchromium.so  
unzip webview.zip  
  
# 预制到Android工程  
cp webview.apk vendor/rockchip/common/webkit/  
cp libwebviewchromium.so vendor/rockchip/common/webkit/
```

修改`/path_to_android/vendor/rockchip/common/webkit/webkit.mk`，具体如下：

```
PRODUCT_COPY_FILES += \  
    vendor/rockchip/common/webkit/webview.apk:system/app/webview/webview.apk \  
    vendor/rockchip/common/webkit/libwebviewchromium.so:system/lib/libwebviewchromiu  
m.so
```

2. For android 6.0 and up

```
# webview_xxx.apk即你要替换的WebView APK  
cp external/chromium-webview/prebuilt/$ARCH/webview.apk
```

3.2 视频无法播放

这里说的视频如果没有特别说明，都是指的 HTML5 Video，无法播放指的是必现的那种，不是随机或特定条件下触发的无法播放。

3.2.1 手势限制

Android 上的 HTML5 Video 默认都要通过手势（触摸和鼠标操作都算）来触发，如果网页直接在一些非手势触发的侦听里直接调用 HTML5 Video 的 play 函数，是会被直接忽略掉的。如果不喜这个特性，可以通过如下代码来关闭：

```
webView.getSettings().setMediaPlaybackRequiresUserGesture(false);
```

如果要让所有调用 Webview 的应用都生效，则可以这样修改 /path_to_android/frameworks/base/core/java/android/webkit/WebView.java：

```
public WebSettings getSettings() {  
    checkThread();  
    mProvider.getSettings().setMediaPlaybackRequiresUserGesture(false);  
    return mProvider.getSettings();  
}
```

3.2.2 安全限制

从某个版本的 Webview 开始，不再允许 HTTPS 和 HTTP 混用，即 HTTPS 的网页是不允许嵌入 HTTP 的内容的，所以就有一些客户反应，Android 6.0 开始的 Webview 无法播放 QQ 视频，认真看一下 logcat 就会发现是视频的 URL 不符合这条安全规则被阻止了。如果不喜这个特性，可以通过如下代码来关闭：

```
webView.getSettings().setMixedContentMode(WebSettings.MIXED_CONTENT_ALWAYS_ALLOW);
```

3.2.3 其他问题

还有一些比较少见的播放限制也会导致无法播放，例如视频源在本地时要保证视频的访问权限，视频在全屏播放的时候需要电源锁的权限，这些都需要 APK 在 Manifest 里正确声明，这个问题 logcat 会抛一个异常出来并且注明是哪个权限失败了；还有就是 GPU 的最大纹理会限制视频的最高分辨率，所以一些老的 GPU 可能会不支持 4k 视频，这个 logcat 也会有明确的提示，并且打开 Chrome，在地址栏输入 chrome://gpu，然后下拉找到 *Video Acceleration Information* 这个表格，里面有详细的视频限制。

3.3 视频卡顿

目前最常见的视频卡顿大致分如下几类：

- 视频走的软件解码，导致受限于 cpu 处理能力

解决办法：

1. 首先要现确认 soc 是否支持这个视频格式的硬件解码，如果是支持的，则需要查一下 Meida 框架层是否哪里判断错误或者没实现，目前 Webview 调用 Media 只有两种方式：MediaPlayer 和 MediaCodec，后者是新版本的默认方式。
2. 如果硬件确实不支持，则可以想办法让网站换个视频格式，目前大部分网站都会通过识别 UserAgent，调用 HTML5 Audio/Video 的[canPlayType\(\)](#)函数等方式来识别本机支持的视频格式。所以这种情况可以先看网站的 Javascript 源码，即使混淆过，大部分判断逻辑还是能猜出的，如果确实有调用 canPlayType 则可以先把当前的视频格式从媒体框架的支持列表中去掉，即 MediaCodecList⁴ 中去掉，可以手写一个简单网页来验证你的修改结果，可在这个[Demo](#)基础上修改。如果 MediaCodecList 修改以后 canPlayType 返回还是不对，那就要查一下整个调用路径了，因为中间代码有大量的 Blacklist 的修改，有兴趣可以看其中一部分[代码](#)。这里就不详解了。

- 主动丢帧，有发现一些应用如爱奇艺会主动丢掉一些帧

解决办法：

1. 只有调用 MediaCodec 或 OMX 的方式解码，应用才有机会主动丢帧，由于第三方应用并没有源码，这时候通常无法知道应用为什么会主动丢帧，最简单的解决方式就是不让应用主动丢帧，比如丢改 MediaCode.c.releaseOutputBuffer(int index, boolean render)，其中 render 位 false 就是代码应用要丢帧，不 care 这个参数即可。
- 内存带宽不足，整个硬件解码流程可以分三个阶段（这里只讲 Webview 的方式）：VPU 解码、缩放裁剪、GPU 贴图，只有中间这个阶段有时候是 CPU 做的（很多时候有 RGA 加速），但全程对内存带宽都有一定要求，所以大部分时候瓶颈都在内存带宽上。

解决办法：

1. 提高内存频率验证效果

3.4 视频无法循环或自动播放

循环和自动播放都是网页通过属性来控制，具体可以参见这两个 Demo：[loop](#) & [autoplay](#)。其他音视频接口和属性完整描述可参见[W3C](#)。

3.5 动画或游戏卡顿

动画和游戏主要有两种实现 CSS 和 HTML5 Canvas，这两种方式都是支持 GPU 硬件加速的，但是 Chromium 有个黑名单机制，会根据 GPU 和各种软件驱动版本来决定是否开启硬件加速，以解决各种兼容性 BUG。所以如果碰到网页动画卡顿，并且更新到新版本也解决不了的时候，可以用 Chrome 打开地址`chrome://gpu`，就会看到当前系统的硬件加速情况，类似如下：

Graphics Feature Status

- Canvas: **Hardware accelerated**
- CheckerImaging: **Disabled**
- Flash: **Hardware accelerated**
- Flash Stage3D: **Hardware accelerated**
- Flash Stage3D Baseline profile: **Hardware accelerated**
- Compositing: **Hardware accelerated**
- Multiple Raster Threads: **Enabled**
- Native GpuMemoryBuffers: **Hardware accelerated**
- Rasterization: **Hardware accelerated**
- Surface Synchronization: **Enabled**
- Video Decode: **Hardware accelerated**
- WebGL: **Hardware accelerated**
- WebGL2: **Hardware accelerated**

关注这些子项即可：Canvas & Rasterization & WebGL & WebGL2，红色即表示关闭，具体的原因可以下拉，关注 Problems Detected，会给出简单的原因描述和 BUG 号，根据这个 BUG 号搜索如下目录：/path_to_chromium/gpu/config，就能找到判定条件，例如：

```
{
  "id": 104,
  "description": "GPU raster broken on PowerVR Rogue",
  "cr_bugs": [436331, 483574, 684586],
  "os": {
    "type": "android"
  },
  "gl_renderer": "PowerVR Rogue.*",
  "driver_version": {
    "op": "<",
    "value": "1.8"
  },
  "features": [
    "accelerated_2d_canvas",
    "gpu_rasterization"
  ]
}
```

上图就表示，PowerVR Rogue 全系列在 Android 系统下，如果 GPU 驱动版本小于 1.8 则关闭 Canvas & Rasterization 两个加速。这时候我们就知道只要升级一下 PowerVR 的驱动到 1.8 以上的版本即可。

3.6 黑屏、白屏和闪烁

碰到这种问题一般先看 logcat 上有没有明确的报错，比如 Graphics Buffer 溢出，GPU 奔溃或 OpenGL 报错，框架层异常等，然后请这些外围模块的工程师协助。当然还有一种情况就是 logcat 找不到明显报错，这时候可以尝试升级 Webview 版本，还不行的话就升级 GPU 驱动版本，大部分情况下这些问题都是 Webview 和图形驱动的兼容问题导致。

3.7 如何修改 UserAgent

修改 UserAgent 目的是为了伪装设备端，比如伪装成 Desktop 或 Ipad 等，可以通过如下代码修改 Webview 的 User-Agent：

```
webview.getSettings().setUserAgentString("");
```

如果要修改 Chrome 的 UserAgent，可以通过如下方式实现：

```
echo 'chrome --user-agent="Mozilla/5.0 (Linux; Android 4.4; Nexus 7 Build/KRT16M) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/30.0.1599.92 Safari/537.36"' > /data/local/chrome-command-line
```

要预制到固件的话，可以修改/path_to_android/vendor/rockchip/common/webkit/webkit.mk，打开注释，类似如下：

```
PRODUCT_COPY_FILES += \
    vendor/rockchip/common/webkit/chrome-command-line:system/etc/chrome-command-line \
    vendor/rockchip/common/webkit/chrome.sh:system/bin/chrome.sh
```

同时修改/path_to_android/vendor/rockchip/common/webkit/chrome-command-line，替换你想要的 UserAgent。

==注意==：修改 UserAgent 可能会导致兼容变差，例如你伪装成 Ipad 的 Safari 浏览器，结果服务器返回的页面就用了 Safari 才能支持的一些特性，那就会出问题了。

3.8 如何实现网址过滤

Android 本身不支持网址过滤功能，要实现这个功能，有三种方式：

- 直接修改/etc/hosts

下面是一个例子，禁止访问百度：

```
127.0.0.1      localhost
::1            ip6-localhost
0.0.0.0        www.baidu.com
```

这个方法很简单，并且是全局有效，但是缺点也很明显：不支持通配符；不支持白名单机制，更新也不方便；通过 socket 编程可以绕过。

- 修改/path_to_chromium/net/url_request/url_request.cc，Webview 的所有请求都会经过这个类，可以在这里去过滤 URL，通过控制 Read 函数是否允许返回数据即可实现管控。

这个方法缺点很明显，只对调用 Webview 的浏览器有效，对于自己实现 Web Engine 的浏览器无效。优点是比较灵活，黑名单和白名单都可以，更新也容易。

- 用 netfilter 和 iptable 做包过滤

这种方法可以实现最灵活也最严格的管控，很多防火墙都是这样实现的，很难绕过。具体方法可以参考这个[教程](#)。

3.9 无法打开下载文件

这个主要是由于服务器给文件设置了错误的 mime-type，例如给 APK 设置了 *text/plain*，这就导致了浏览器报给下载管理器的类型错了，会导致直接通过消息和下载管理器打开会失败，通过文件管理器可以正确打开。要修正这个问题可以这样修改：

```
commit c906b4b4f9a120d28a099fece4a5820127a32201
Author: mouchun chen <cmc@rock-chips.com>
Date:   Wed Jan 7 18:09:29 2015 +0800

fix wrong mimetype when download file

diff --git a/src/com/android/browser/DownloadHandler.java
b/src/com/android/browser/DownloadHandler.java
index 7a24aa4..8c1441d 100755
--- a/src/com/android/browser/DownloadHandler.java
+++ b/src/com/android/browser/DownloadHandler.java
@@ -222,7 +222,9 @@ public class DownloadHandler {
        Log.d(LOGTAG, "referer: " + referer);
        request.setNotificationVisibility(
            DownloadManager.Request.VISIBILITY_VISIBLE_NOTIFY_COMPLETED);
-       if (mimetype == null) {
+       if (mimetype == null
+           || mimetype.equalsIgnoreCase("text/plain")
+           || mimetype.equalsIgnoreCase("application/octet-
stream")) {
            if (TextUtils.isEmpty(addressString)) {
                return;
            }
        }
    }
```

3.10 是否支持 Adobe Flash

Adobe 从 Android 4.3 版本开始停止了对 Android 上 Flash 插件的支持，我们最后可支持插件的版本是 Android 5.1。目前的替代方案有两个：Adobe AIR 和 HTML5，新的内容可以用这两个来实现，这里更推荐后者。而原来旧的 Flash 内容，如果有源文件（即 fla 文件），可以通过新版本的 Adobe Flash 开发工具直接倒出成两个替代方案。

3.11 崩溃和 ANR 如何处理

对于这两类问题，在丢到我这边来处理前，我希望能做到这几点，来尽量提高处理效率。

- 先自己简单过一遍 log 信息

对于崩溃，可以先看一下 logcat，到底死在哪，如果是外围模块比如框架层、Media 和 Graphics，可以直接转给相关模块工程师。如果是死在浏览器内部，则可以转给我，并且如果是死在 Native 层需要先提供一下 Crash Dump 的符号表，方式如下：

```
addr2line -e out/target/product/rk3399/symbols/system/lib/libc.so 2000
```

其中 libc.so 要替换成死掉的那个库，2000 替换成死的位置，这些都能在 Crash Dump 里找到。

对于 ANR，要同时提供 logcat 和 /data/anr/traces.txt，并且先过一下 traces.txt 看主线程是堵在哪里，如果确认是外围模块，同样可以直接转给相关工程师。

- 有条件的话，试一下其他方案有没有类似问题
- 试一下新版本的 Chrome 或 Webview 能否解决这个问题

==注意==：Android 6.0 开始 Webview 也是预制 APK 形式发布，所以也没有符号表了，如果崩溃在 libwebviewchromium.so 就没法 debug，而 Chrome 一直都是 APK 发布，但是其崩溃会自动给 Google 发错误信息，Google 会有选择的抽一些来看。

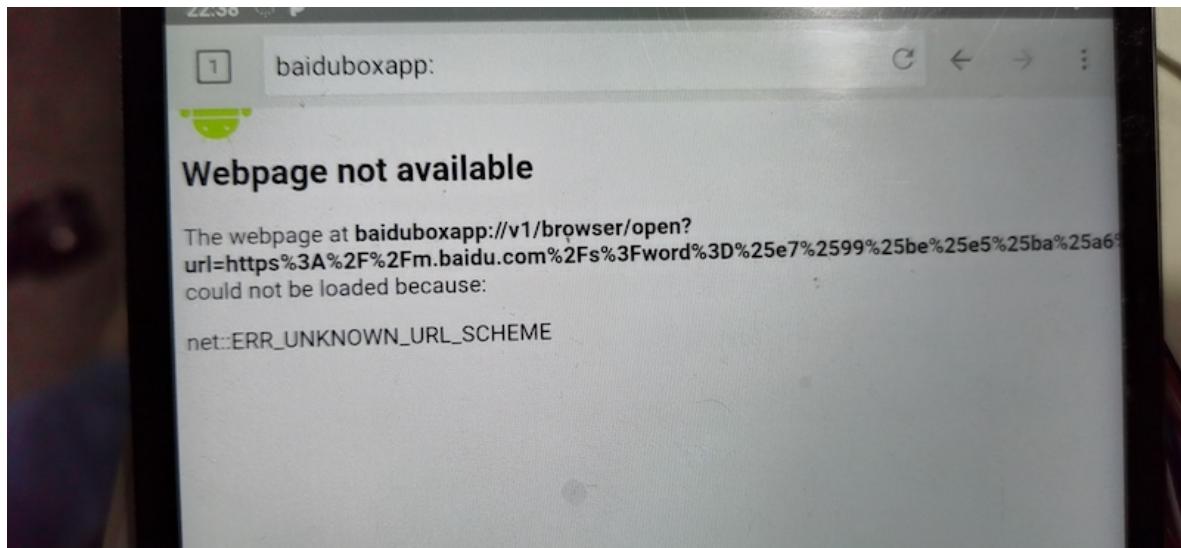
3.12 视频有声无影

这种问题都可以先试一下这样修改能否解决：

```
# setprop sys.hwc.compose_policy 0  
# stop  
# start
```

3.13 浏览器上百度首页，会报错： net::err_unknown_url_scheme

这个问题是百度导致的，百度的服务器会识别移动设备，在首次访问百度的时候尝试跳转到百度自己的应用，而机器没有安装百度的应用，识别不了 baiduboxapp 这个 scheme，进而导致访问报错。此时浏览器会显示如下错误：



因为只有首次访问才会有这个提示，所以这个错误可以直接忽略，如果实在要解的话，只能参考前面章节介绍的方法修改 `UserAgent`，目前已知 `Desktop` 的 `UserAgent` 是有效的，但是会导致一些网页被显示成桌面模式。同时猜测一些手机厂商的 `UserAgent` 应该也是可以的。

3.14 所有调用 webview 的浏览器都闪退，logcat 提示 java.lang.ClassNotFoundException: Invalid name:

这个问题的原因是因为 RTC 出异常，同时 NTP 服务出错（NTP 服务器地址不对，或网络不通），导致系统的时间错的很厉害，超过数字证书的有效时间范围，所以 APK 的签名校验失败，进而在虚拟机加载资源的时候，会出现 Class Name 为空的情况，最后虚拟机崩溃退出。这个问题还有如下特征：

- 那些不用webview的浏览器虽然能够打开，例如：Chrome和Firefox，但是它们打开绝大部分网站都会提示数字证书检查失败，基本都会弹出一个对话框，提示你网站有风险。
- 很多APP的安装可能出现问题，特别是那些本来数字证书有效期就不长的APP。
- 因为是RTC的问题，问题还经常跟着机器走，即使重烧系统也无法回复。
- ADB shell敲 date 命令可以发现当前系统时间误差有好几年。
- 通过 ADB shell 敲 `date "YYYY-MM-DD [hh:mm[:ss]]"` 命令来修正当前时间，并重装webview（参见本文档相关章节），问题就消失了。

解决方案：

这个问题要从根本上解决，还得去修复RTC问题，手动通过 date 修正，重启以后基本上又会浮现。

4. How to build Webview

有兴趣可以参见 Google 的[说明文档](#)。需要注意的是，Chromium 的提交非常频繁，所以最新分支稳定性不保证，甚至不一定能编译过，如果要出固件的话最好是基于稳定分支 LKGR，或则基于指定的稳定版本，可以参考最新版本的 Chrome，通过地址栏输入`chrome://version`可以看到版本。切换指定版本可以参考这个[说明文档](#)。

5. How to debug

有时候某些网页元素显示异常，需要看网页的 Layout 信息，在旧版本的 Android(4.2 & before)，可以通过在地址栏输入`about:debug`等特殊地址来启动 dump，Layout 信息会以文本形式输出在/sdcard 目录。新版本的 Webview 没有类似的调试手段，但是 Chrome 可以实现远程调试，具体步骤参见[Google 文档](#)。

而还有一种情况，APK 调用 Webview 或其他 Web Engine 出现显示异常，这时候在设备端没有太好的调试手段，这时候又没有网页的作者配合查的话就很麻烦。可以尝试如下办法：

- 如果有对比方案没问题

可以从两个方面来排查，Android 版本是否不一样，可以切到一样的版本对比看看，如果 Android 版本不好切，可以切 Webview 版本看看（如果有调用 Webview 的情况下）；UserAgent 是否有差异，可以通过 `tcpdump` 来抓包对比，尝试修改 UserAgent 看能否解决。

- 用设备的 UserAgent，通过桌面版的 Chrome 的开发工具（设置->更多工具->开发者工具），弹出的工具栏有一个类似手机的按钮，里面有设备模拟功能，模拟我们的设备端加载同一个网页，然后就能通过 Elements 选项卡找到显示异常的元素，至少能大幅缩小范围，最后看服务器端能否换一个写法来实现这个元素。

6. 重装webview

前面介绍的升级和集成webview的方式比较繁琐，在调试过程中不太好用，在调试过程中可以通过如下方式来先卸载当前的webview：

```
adb root  
adb remount  
rm -rf /product/app/webview  
sync  
reboot
```

重启完成后，`webview`就会被自动卸载掉，此时就可以通过`adb install`命令来自由安装你的`webview`了，测试通过以后再考虑集成到系统固件中。

1. 通过 Google Play 来更新[2](#)
2. 只支持 Android 7.0 & after[2](#)
3. 这是 GMS 包里默认的 Webview，需要过 Google GMS 认证才能集成[2](#)
4. /path_to_android/frameworks/base/media/java/android/media/MediaCodecList.java[2](#)