

目录

前言	1.1
Android开发概览	1.2
Android设备	1.3
Android设备类型	1.3.1
Android版本	1.4
Android环境变量	1.5
Android Studio	1.6
AS智能之处	1.6.1
Android常用库	1.7
Android常用代码段	1.8
附录	1.9
参考资料	1.9.1

Android开发总结

- 最新版本: v0.7
- 更新时间: 20230724

简介

总结Android开发的心得。先是概览；然后介绍Android设备和设备类型；介绍版本号；环境变量；开发语言；以及目前用的最多的IDE即Android Studio，智能之处；以及常用的库；常用代码段。

源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

HonKit源码

- [crifan/android_dev_summary: Android开发总结](#)

如何使用此HonKit源码去生成发布为电子书

详见：[crifan/honkit_template: demo how to use crifan honkit template and demo](#)

在线浏览

- [Android开发总结 book.crifan.org](#)
- [Android开发总结 crifan.github.io](#)

离线下载阅读

- [Android开发总结 PDF](#)
- [Android开发总结 ePUB](#)
- [Android开发总结 Mobi](#)

版权和用途说明

此电子书教程的全部内容，如无特别说明，均为本人原创。其中部分内容参考自网络，均已备注了出处。如发现有侵权，请通过邮箱联系我 [admin 艾特 crifan.com](mailto:admin@crifan.com)，我会尽快删除。谢谢合作。

各种技术类教程，仅作为学习和研究使用。请勿用于任何非法用途。如有非法用途，均与本人无关。

鸣谢

感谢我的老婆陈雪的包容理解和悉心照料，才使得我 [crifan](#) 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

其他

作者的其他电子书

本人 crifan 还写了其他 150+ 本电子书教程，感兴趣可移步至：

[crifan/crifan_ebook_readme: Crifan的电子书的使用说明](#)

关于作者

关于作者更多介绍，详见：

[关于CrifanLi李茂 – 在路上](#)

crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新：2023-07-24 22:17:25

Android开发概览

有折腾过Android开发，整理一些相关的心得。

- Android开发语言
 - 早期: Java
 - 最新: Kotlin
- Android开发工具
 - 早期: Eclipse+ADT
 - 最新: Android Studio

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2022-11-03 18:09:10

Android设备

TODO:

- 【已解决】安卓手机中查看app应用信息中的包名信息

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2022-11-03 17:40:57

Android设备类型

Google 只是研发了 Android 的移动设备的操作系统，而普通用户买到的手机，都是各个厂商，比如 小米 、 华为 、 OPPO 等，加上硬件和研发最终设计出来的各自品牌的手机。

也因此，Android的设备类型，相对iOS的有限的设备来说，就显得多而杂，有高端有低端，有大屏有小屏，而且各自等屏幕尺寸又不一定完全相同。

Android的常见设备列表

从[网上](#)，整理出 Android 的设备列表，供参考：

- 小米
 - 小米3C/小米3W
 - 小米4C/小米4W/小米4i/小米4S
 - 小米5/小米5S//小米5SPlus
 - 小米MIX/小米MAX
 - 小米Note2/小米Note Pro//小米Note LTE
 - 小米平板/小米平板2
 - 红米1S
 - 红米3S/红米3X
 - 红米4/红米4A
 - 红米Note1/红米Note3 全网通版
- OPPO
 - A30/A57/A59m/A59s
 - Find7(X9077/X9007/X9000/X9070)
 - N3(N5207/N5209)
 - R3(R7005)
 - R9(R9km/R9m/R9s/R9t/R9tm/X9009)
 - R9 Plus(R9 PlusmA/R9 Plust A/R9 Plusm A)
 - R9s(R9st/R9sk)
 - R9s Plus
 - VIVOX3 V
 - X5 M/X5 M L/X5 Max L/X5 Max+/X5 Pro L/X5 Pro V
 - X6 D/X6 L/X6 Plus D/X6 Plus L/X6 S/X6 SA/X6 S Plus/X6 S Plus A/X6 S Plus D
 - X7/X7 L/X7 10F/X7 10L/X7 Plus/X7 Plus L
 - X9
 - Xplay3S(X520F)/Xplay5A/Xplay5S/Xplay6
 - Y55A/Y67
- 魅族
 - M5 Note/MX4/MX4 Pro/MX4 YUNOS版/MX5/MX6
 - Pro5/Pro6/Pro6 Plus/Pro6s
 - 魅蓝E/魅蓝Metal/魅蓝M3 Max/魅蓝Note/魅蓝Note2 公开版 移动定制版 联通定制版/魅蓝Note3/魅蓝U20
- 华为
 - 手机
 - G9青春版(VNS-DL00/ VNS-AL00/VNS-TL00)/G9 Plus(MLA-TL00/MLA-UL00/ MLA-TL10)
 - Nova(CAZ-AL10/ CAZ-TL10)
 - Mate 7(CL00/UL00/TL00/TL10/L09)
 - Mate 8(AL10/CL00/DL00/TL00/eH880/NXT-L29)
 - Mate 9(MHA-AL00)/Mate 9 Pro(LON-AL00)
 - Mate S(CRR-CL00/CRR-UL00/CRR-TL00/CRR-CL20/CRR-UL20)

- P8(GRA-CL00/GRA-CL10/GRA-TL00/GRA-UL00/GRA-UL10)/P8 Max
- P9(EVA-AL00/EVA-AL10/EVA-CL00/EVA-DL00/EVA-TL00/EVA-L19)
- P9 Plus(VIE-L29/VIE-AL10)
- 麦芒5(MLA-AL00/MLA-AL10)
- 荣耀6Plus(PE-UL00/PE-TL20/PE-TL10/PE-TL00M/PE-CL00)
- 荣耀6(H60-L02/H60-L11/H60-L01/H60-L03/H60-L12/H60-L21)
- 荣耀7(PLK-CL00/PLK-AL10/PLK-TL01H/PLK-UL00/PLK-TL00)
- 荣耀8(FRD-AL00/FRD-AL10/FRD-DL00)/荣耀 Note8(EDI-AL10)
- 荣耀X2(GEM-703L/GEM-703LT)
- 荣耀V8(KNT-AL10/KNT-AL20/KNT-TL10/KNT-UL10)
- 荣耀畅玩5A(CAM-AL00)
- 荣耀畅玩5C(NEM-AL10/NEM-TL00H/NEM-TL00/NEM-UL10)
- 荣耀畅玩6X(BLN-AL10/BLN-TL10)

- 平板

- (M2-A01L/M2-A01w/M3 BTV-DL09/M3 BTV-W09)

- 三星

- A5(A5100/A5108)/A7(A7100/A7108)/A9(A9000/A9100)/Alpha(G8508S)
 - C5(C5000)/C7(C7000)/C9 Pro(C9000)
 - J(SC-02F)
 - Note 3(N7505/N7509V/N9007/N900J/N900K/N900L/N900P/N900V/SC-01F/N900/N9002/N9005/N9006/N9008/N9008S/N9008V/N9009/N 900S)
 - Note 3 Lite(N7506V/N7508V)
 - Note 4(N910C/N910F/N910G/N910K/N910L/N910P/N910S/N910U/N9100/N9106W/N9108V/N9109W)
 - Note 5(N9208/N920C/ N920K/N920L/N920P/N920S/N920V/N9200)
 - Note 7(N9300)
 - Note Edge(N9150/N915S)
 - S4(I9500/I9502/I9505/I9507/I9507V/I9508V/L720/SCH-I959)
 - S4 LTE-A(E330S)
 - S5(G9006V/G9006W/G9008V/G9008W/G9009D/G9009W/G900F/G900H/G900I/G900J/G900K/G900L/G900P/G900V/G906K/SCL23)
 - S5 LTE-A(G906L/G906S)
 - S6(G9200)G9208/G9209/G920F/G920k/G920L/G920P/G920S/G920T/G920V/SC-05G)
 - S6 Edge(G920I/G9250/G9250L/G925A/G925F/G925I/G925K/G925P/G925S/G925T/G925V/SCV31)
 - S6 Edge+(G9280/G9287/G928A/G928F/G928K/G928P/G928S/G928T/G928V)
 - S7(G9300/G9308/G930F/G930K/G930L/G930S/G930T/G930U/G930V)
 - S7 Edge(G9350/G935F/G935K/G935L/G935S/G935T/G935u/G935V)
 - W2015/W2016
 - Galaxy 旗舰III(G9198)

- 乐视

- 乐 Pro3(X720)
 - 乐1 Pro (X800/ X800+)
 - 乐2(X520/X528/X620/X621)/乐2 Pro(X625)
 - 乐Max(X900/X900+)/乐Max 2(X820/X822)/乐Max Pro(X910)

- 一加

- 一加手机1 (A0001/A1001)
 - 一加手机2(A2001/One A2001)
 - 一加手机3(OnePlus A3000)
 - 一加手机3T (A3010)
 - 一加手机X (ONE E1001/ONE E1000)

- 360

- N4(1503-A01/503-M02)/N4A(1603-A03)/N4S(1505-A01/1505-A02)
 - Q4(1509-A00)/Q4青春版(1515-A01)

- 努比亚

- nubia N1(NX541J)
 - nubia X6(NX601J)
 - nubia Z5s(NX503A)
 - nubia Z7 Max(NX505J)/nubia Z7 Mini(NX507J)/nubia Z7(NX506J)
 - nubia Z9 Max(NX510J/NX518J/NX508J)
 - nubia Z11(NX531J)/nubia Z11Max(NX523J_V1)/nubia Z11Mini(NX529J)/nubia Z11Minis(NX549J)/nubia z11 Max*(NX535J)
- 中兴
 - Axon Mini(B2015)/Axon天机(A2015)/Axon天机7(A2017)/Axon天机 Max(C2016/C2017)
 - Blade(BA910/BA910T)/Blade A1(C880U/C880A)/Blade A2 Plus(BV0730)
 - Geek II/Grand S II LTE
 - V5(N9180/X9180)
 - V7 MAX(BV0710/BV0710T)
 - 青漾2S(G718C)
 - 小鲜3(C880S/C880D)小鲜4(BV0701)
 - 星星1号(S2002)/星星2号(S2005/S2010)
 - 远航3(Q529C/Q529T)
- 联想
 - Z1(全网通版)/Z2(Z2131)/Z2 Pro(Z2121)
 - S860e(电信版)
 - 乐檬X3(X3c70/K51c78)/乐檬X3青春版(K52t38)/乐檬K5 Note(K52e78)
 - VIBE P2(P2c72)/VIBE X2(X2-TO)/VIBE Z(K910)/VIBE Z2 Pro(K920/X2Pt5)
- 锤子
 - 坚果/坚果Pro/坚果Pro 2
 - M1/M1 L(SM919)
 - T1(SM701/SM705)/T2(SM801)
- 酷派
 - 酷派C106/C106-9/C106-6/C106-8/派C107-9
 - 锋尚MAX(A8-831/A8-930/A8-932/A8-931N)
 - 大观铂顿 V1-C电信版
- 金立
 - 金立
 - E8移动联通版(GN9008)
 - GN715移动联通版
 - M6(GN8003)/M6 Plus(GN8002/GN8002S)
 - S5.1移动版(GN9005)
 - S6 Pro(GN9012)
 - S7(GN9006)/S7 Mini(GN9007)
 - S8 全网通(GN9011)
 - 大金钢全网通(GN5003)
 - IUNI
 - IUNI N1
 - IUNI i1
 - IUNI U2
 - IUNI U3
- 索尼
 - Xperia C3(S55t/S55u)
 - Xperia T2 Ultra(XM50h/Xm50t)
 - Xperia X (F5122/F5321)
 - Xperia XP(F8132)
 - Xperia XZ(F8332)
 - Xperia Z Ultra(XL39h)
 - Xperia Z1(C6902/C6903/D5503/L39h)/Xperia Z1 mini (M51w)

- Xperia Z2(L50t/L50u/D6503)
- Xperia Z3(D6633/D6683/D6708/D6603/D6653/D5803/D5833/L55u/L55t/SO-01G/SO-02G/SOL26)
- Xperia Z3+ dual (E6553/E6533)
- Xperia Z5(E6653/E6683)/Xperia Z5 Compact(E5823)/Xperia Z5 Premium(E6883/E6853)
- 摩托罗拉
 - Moto Z(XT1650-05)/Moto Z Play (XT1635-03)
 - Moto M(XT1662)
 - Moto X(XT1060)
 - LGAKA(H778)
 - G Flex(D958/LS996)
 - G Pro 2 (LG-D838)
 - G2(D802/L22/LS980/VS980)
 - G3(D855/D857/D858/D858HK/D859/F400L/F460/F460K/F460S/L24/LS990/VS985)
 - G4(F500L/H815/H818/H819)
 - G5(F700L/H850/H860/H868/LS992)/G5 SE(H848)
 - V10(F600L/F600S/H901/H961N/H968/VS990)
- 谷歌
 - Nexus 5/Nexus 5X/Nexus 6/Nexus 6P
 - Pixel/Pixel xl
 - HTC10(M10h/M10u)
 - Desire(D830u/D10w)
 - One A9/One A9移动联通版
 - One E9(E9t/E9w)/One E9+(E9pw)
 - One M8si
 - One M9(M9u/M9w/m9e)/One M9+(M9pt/M9pw)
 - One ME(M9et/(M9ew)
 - One X9u
- 中国移动
 - A2(M636/双4G)
 - M812C
 - N2(M836)
- 华硕
 - ZenFone2(ASUS_Z00ADB/ASUS_Z00AD/ASUS_Z00ADA)
 - ZenFone Zoom(ASUS_Z00XSB)
- 海信
 - A1/F3 Pro(F31)/K8(H910)
 - 小岛(E20T)/金盾(E76)
- Gigaset金阶
 - Gigaset ME(GS55-6)
 - Gigaset ME Pro (GS57-6)
- 8848
 - M3
- IVVI
 - IVVI i3(i3-01/SS2-01)
- AGM
 - X1
- PPTV
 - PPTV Mipo M1(PP5600)
- SUGAR
 - SUGAR F7
- TCL
 - 580/750/950
- YotaPhone

- YotaPhone2 (YD201/YD206)
- 泛泰
 - A890S/A910S
- 飞利浦
 - 杰智I999
- 富可视
 - M2/蓝鲸S1
- 格力
 - 格力手机2代 (G0215D)
- 黑莓
 - STV100-1/STV100-3
- 亚马逊
 - Fire Phone(SD4930UR)

crifan.org, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2022-11-03 17:40:35

Android版本

Android的： sdk的int值 = api级别 和 代号 = 版本号 之间关系=映射表

代号=Codename	版本=Version	API 级别=level / NDK 版本=release
Android13	13	API 级别 33
Android12L	12	API 级别 32
Android12	12	API 级别 31
Android11	11	API 级别 30
Android10	10	API 级别 29
Pie	9	API 级别 28
Oreo	8.1.0	API 级别 27
Oreo	8.0.0	API 级别 26
Nougat	7.1	API 级别 25
Nougat	7.0	API 级别 24
Marshmallow	6.0	API 级别 23
Lollipop	5.1	API 级别 22
Lollipop	5.0	API 级别 21
KitKat	4.4-4.4.4	API 级别 19
Jelly Bean	4.3.x	API 级别 18
Jelly Bean	4.2.x	API 级别 17
Jelly Bean	4.1.x	API 级别 16
Ice Cream Sandwich	4.0.3-4.0.4	API 级别 15, NDK 8
Ice Cream Sandwich	4.0.1-4.0.2	API 级别 14, NDK 7
Honeycomb	3.2.x	API 级别 13
Honeycomb	3.1	API 级别 12, NDK 6
Honeycomb	3.0	API 级别 11
Gingerbread	2.3.3-2.3.7	API 级别 10
Gingerbread	2.3-2.3.2	API 级别 9, NDK 5
Froyo	2.2.x	API 级别 8, NDK 4
Eclair	2.1	API 级别 7, NDK 3
Eclair	2.0.1	API 级别 6
Eclair	2.0	API 级别 5
Donut	1.6	API 级别 4, NDK 2
Cupcake	1.5	API 级别 3, NDK 1
(无代号)	1.1	API 级别 2
(无代号)	1.0	API 级别 1

此处列举出自己需要用到的几个：

- Android 6.0 = Marshmallow = sdk的int值即 API 23
- Android 8.1.0 = Oreo = sdk的int值即 API 27
- Android 9.0 = Pie = sdk的int值即 API 28
- Android 10.0 = Android10 = sdk的int值即 API 29
- Android 11.0 = Android11 = sdk的int值即 API 30

crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新：2023-07-24 21:47:56

Android环境变量

- Android相关环境变量
 - 旧的: `ANDROID_HOME`
 - 已废弃
 - 为了兼容, 目前也还支持 `ANDROID_HOME` 而已
 - 新的改用: `ANDROID_SDK_ROOT`
 - 建议用新的

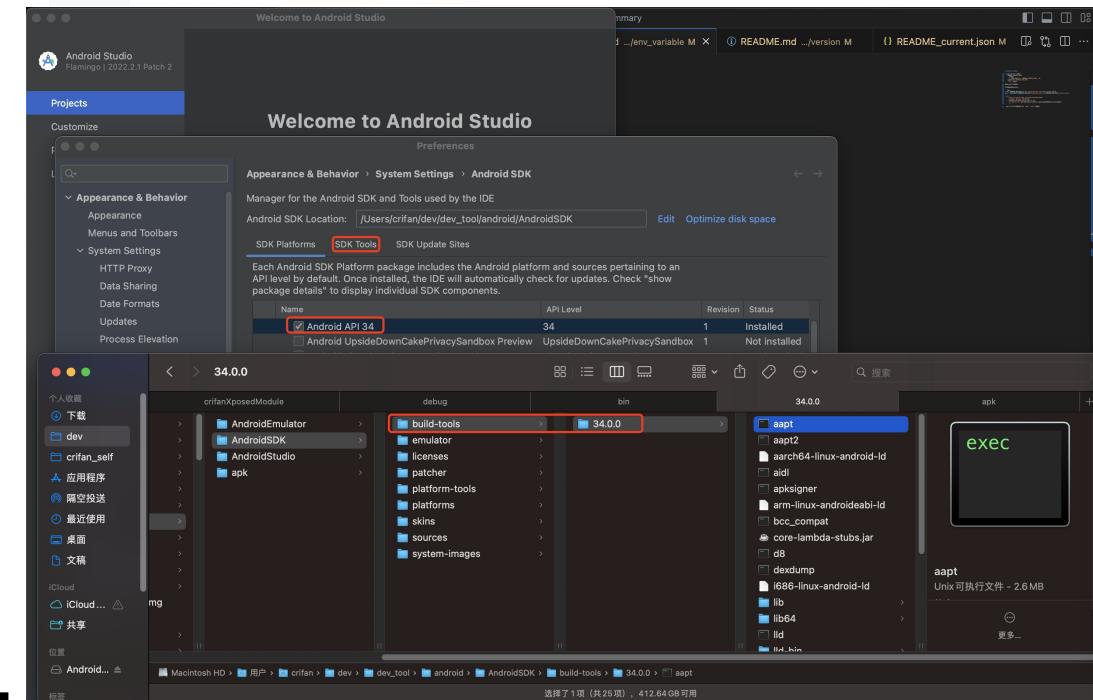
如何设置环境变量:

把下面的变量定义:

```
export ANDROID_SDK_ROOT /Users/crifan/dev/dev_tool/android/AndroidSDK/
export ANDROID_HOME $ANDROID_SDK_ROOT
export PATH $PATH:$ANDROID_SDK_ROOT/build-tools/34.0.0:$ANDROID_SDK_ROOT/platform-tools
```

- 其中

- `/Users/crifan/dev/dev_tool/android/AndroidSDK/`
 - 是我当前 Android SDK 的位置
- `$ANDROID_SDK_ROOT/build-tools/34.0.0`
 - 其中的 34.0.0 是对应的, 已下载好的, 当前Android的SDK的Build-tools的版本



加到启动脚本, 比如此处 Mac 中的 `~/.zshrc` 即可。

crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook 最后更新: 2023-07-24 22:00:44

Android Studio

TODO:

把相关多个帖子内容整理过来

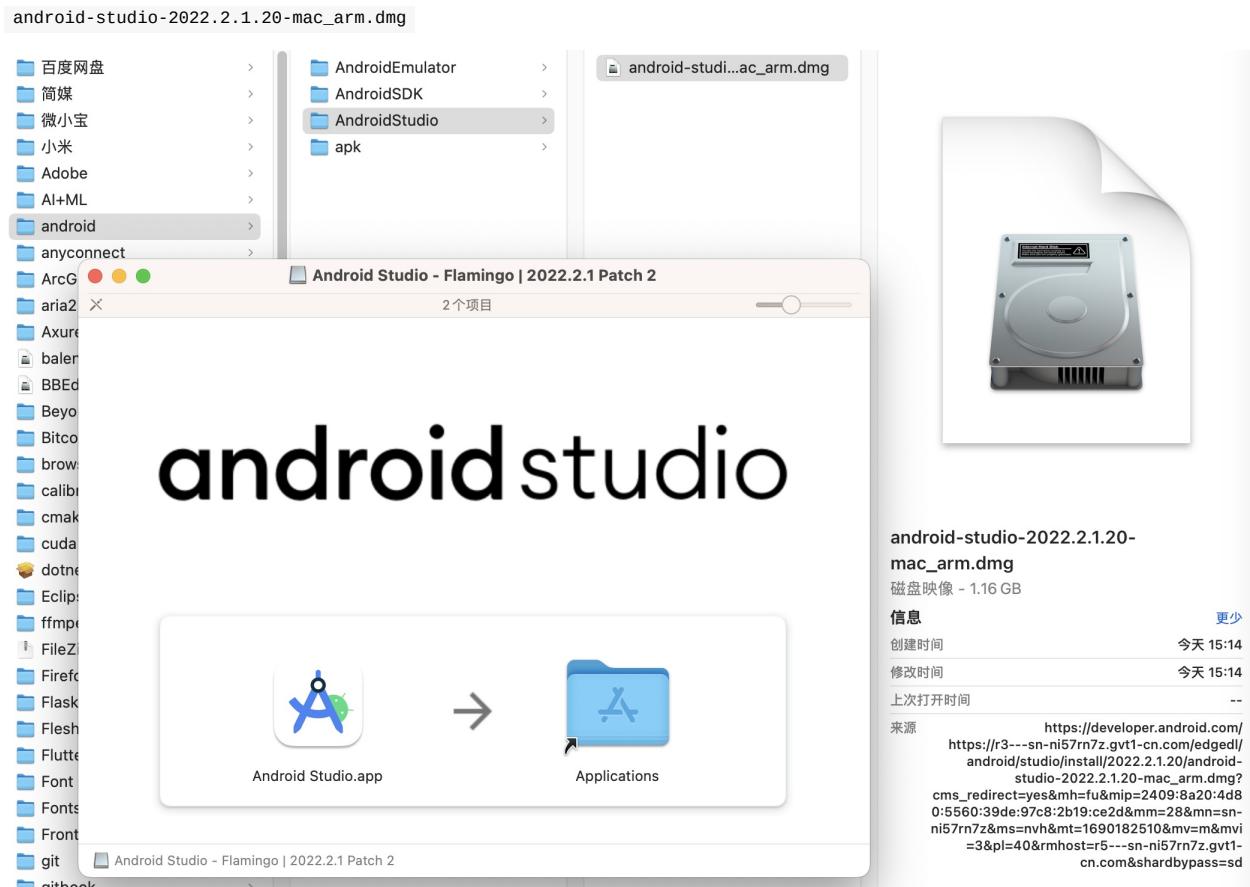
- 【已解决】Android Studio中左边项目文件列表如何跳转切换到当前打开的文件
- 【已解决】Android中用preference-v7后编译报错：values-v26.xml AAPT error resource android attr colorError not found
- 【已解决】Android中context的startService对应的service运行机制
- 【未解决】通过Android Studio打开调试设置Java Exception Breakpoints和Exception Breakpoints看看能否找到rcsita中rcs的core的服务如何启动的逻辑
- 【记录】Android Studio根据提示升级到最新4.0.1版本和更新其他内容
- 【已解决】rcsita的RI运行报错：java.lang.SecurityException Not allowed to bind to service Intent

- Android Studio
 - 简称： AS

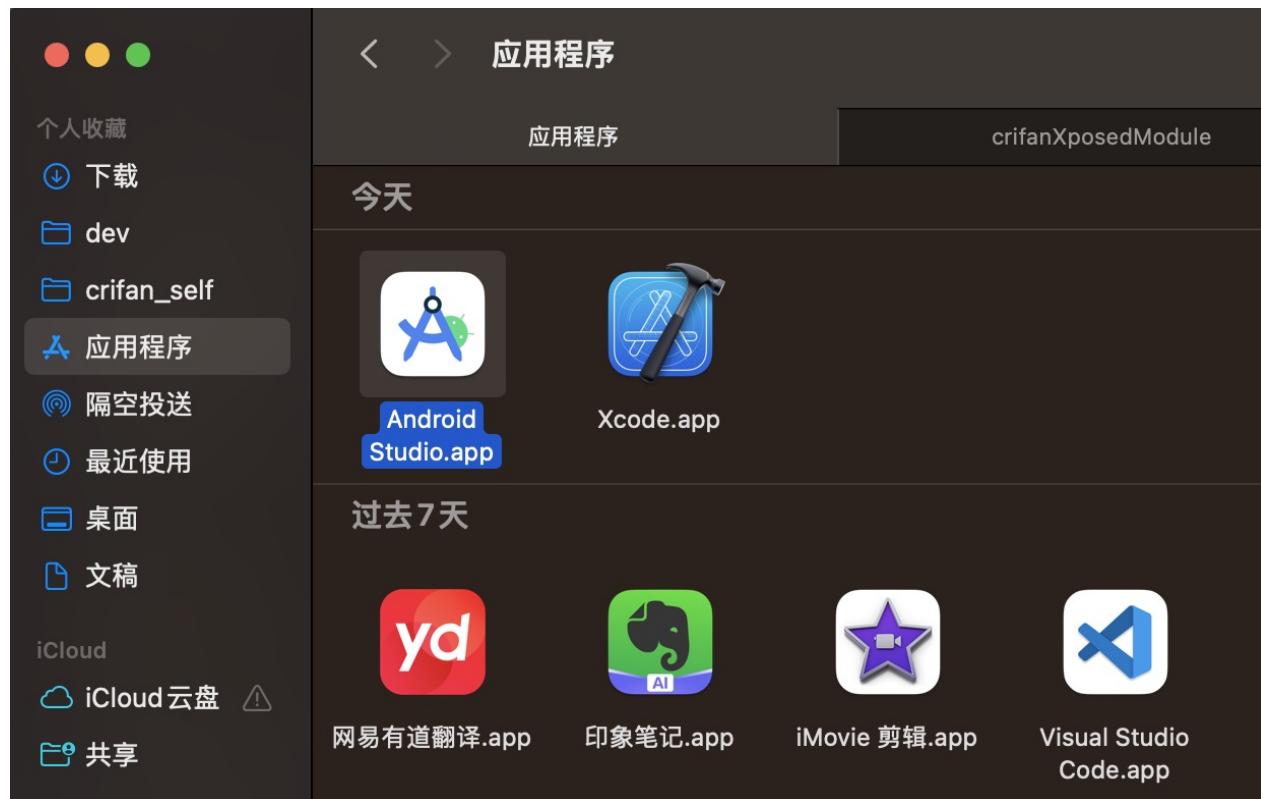
下载和安装

从Android Studio的官网的[下载页面](#)去下载到和当前电脑匹配的安装包，去安装即可。

比如，此处 Mac M2 Max 匹配的是 Apple Silicon 的 Android Studio Flamingo | 2022.2.1 Patch 2 的安装包：

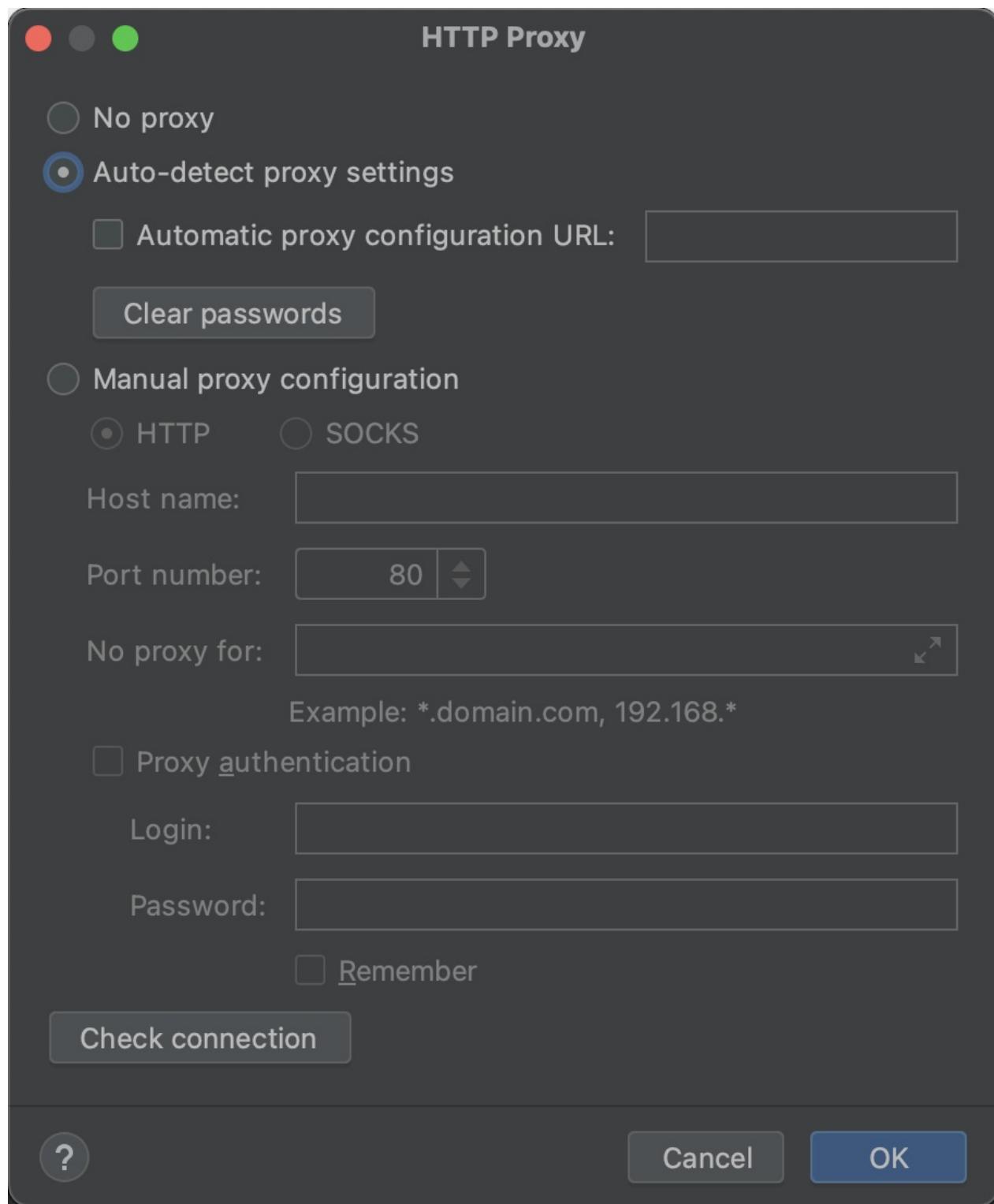


双击后，弹框中，把 Android Studio.app 拖动到 应用程序，即可实现安装：



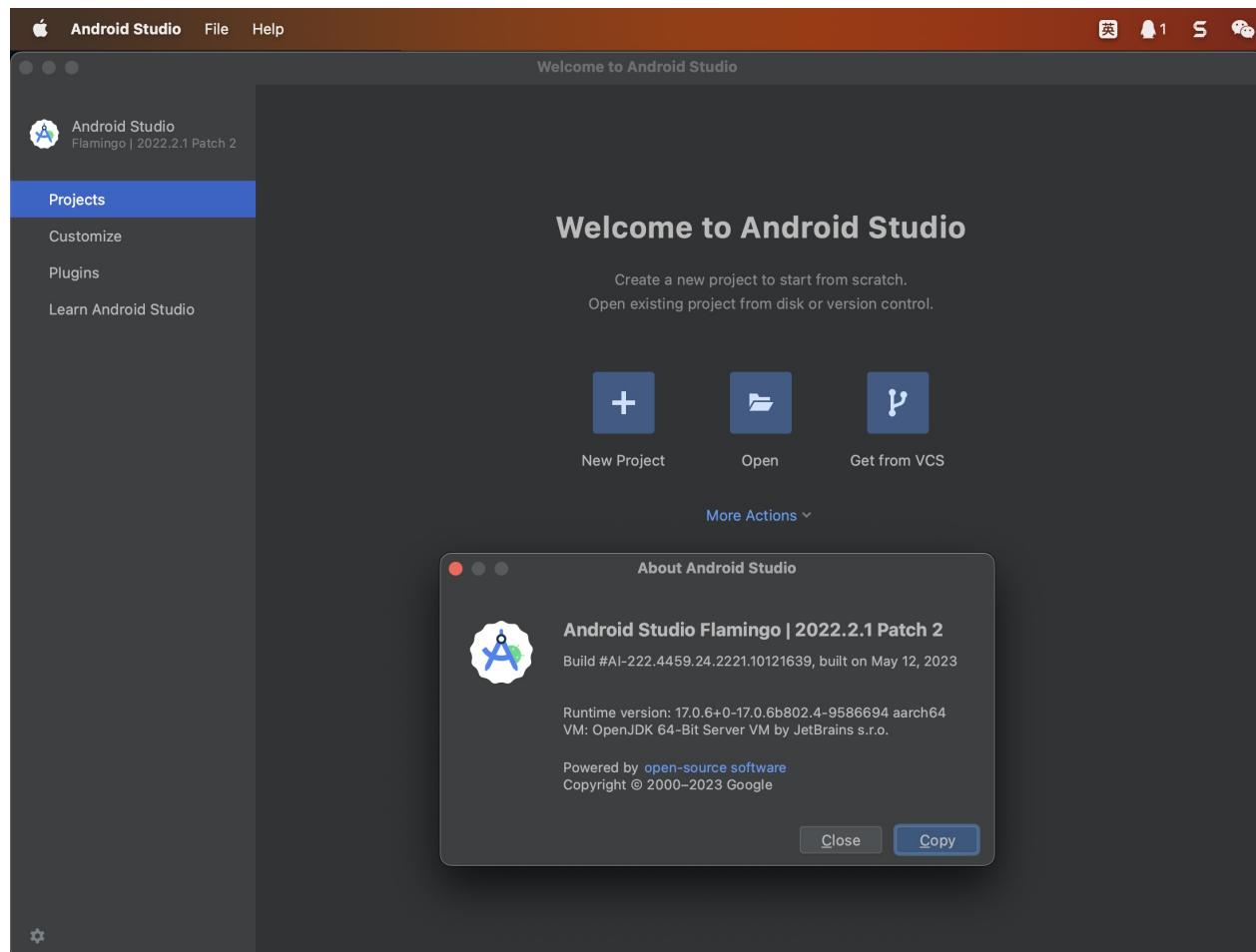
之后即可双击应用程序中的 `Android Studio.app` 启动 AS，双击启动，进行初始化配置

期间需要注意的一点是，最好是，已有代理工具，且此处 `HTTP Proxy` 从默认的 `No Proxy` 改为 `Auto-detect proxy settings`：



这样可以加速后续的google资源的下载。

初始化配置后，即可进入主页面：



此处相关版本信息是：

```

Android Studio Flamingo | 2022.2.1 Patch 2
Build #AI-222.4459.24.2221.10121639, built on May 12, 2023
Runtime version: 17.0.6+0-17.0.6b802.4-9586694 aarch64
VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o.
macOS 13.2.1
GC: G1 Young Generation, G1 Old Generation
Memory: 1280M
Cores: 12
Metal Rendering is ON
Registry:
  external.system.auto.import.disabled=true
  ide.text.editor.with.preview.show.floating.toolbar=false
  gradle.version.catalogs.dynamic.support=true

Non-Bundled Plugins:
  com.developerphil.adbidea (1.6.0)

```

AS快捷键

- 跳转到文件开始/末尾
 - Win : Ctrl + Home / End
 - Mac : Ctrl + Fn + Left / Right
 - 注: Mac 中没有 Home / End
 - 有等价的: Fn + Left / Right
 - Fn + Left = Home
 - Fn + Right = End

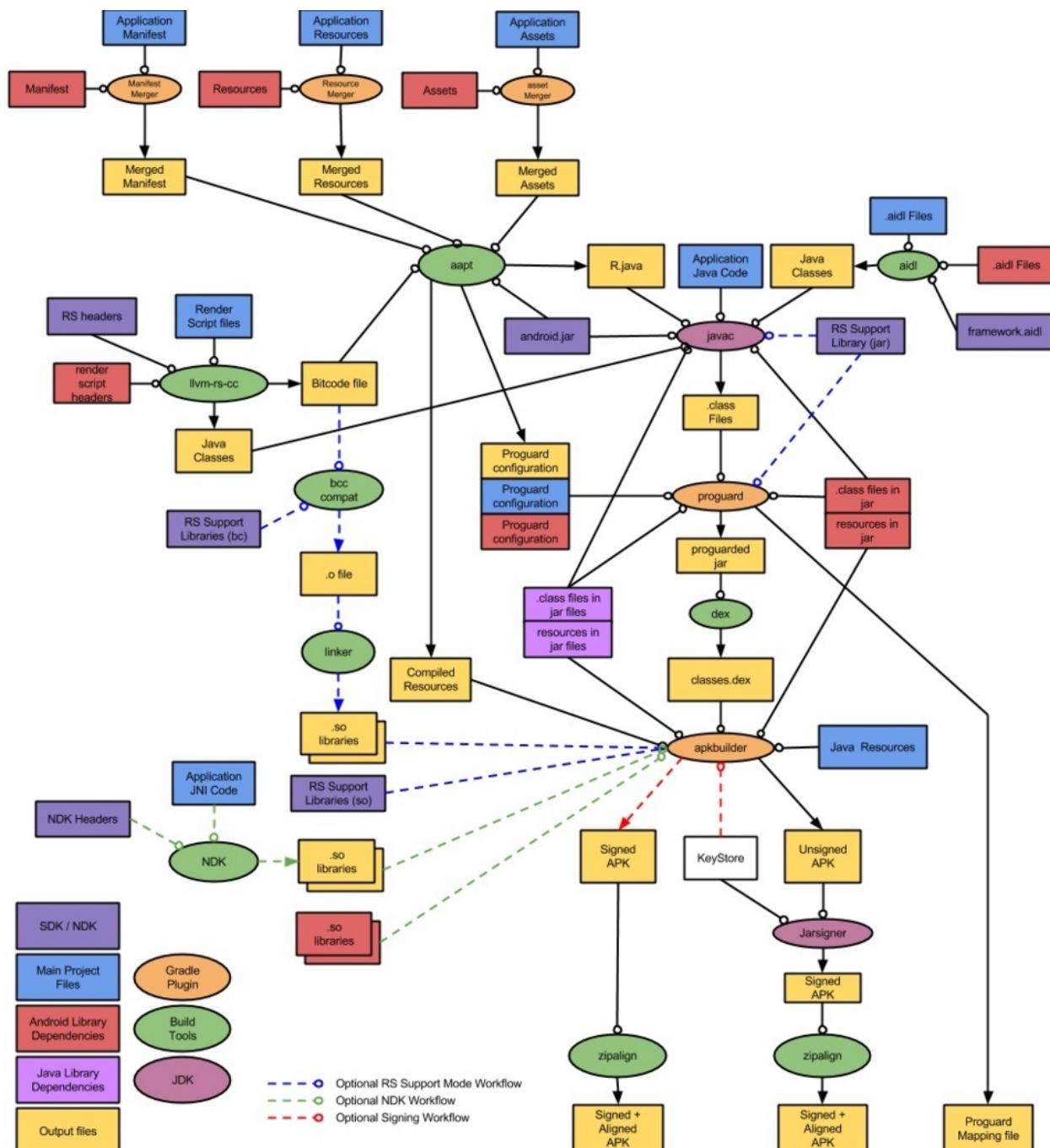
gradle

TODO:

- 【已解决】Mac中Android Studio下载gradle的zip包速度太慢
- 【已解决】Mac中Android Studio中gradle的库下载速度慢
- 【已解决】Android Studio中gradle编译的项目如何加参数: -Xlint:unchecked
- 【已解决】Android Studio中gradle报错: ERROR Cause compileSdkVersion is not specified
- 【已解决】Android中gradle编译报错: This project uses AndroidX dependencies, but the android.useAndroidX property is not enabled

自动化编译过程

安卓编译过程:



-» 手动去操作，很复杂

-》所以一般会需要 build = 构建 工具实现自动化编译过程

安卓的构建工具有

- 最早的： Ant
- 后来的=现在主流的： Gradle
- 最新的： Buck
 - 来自 Facebook

开发安卓的IDE是： Android Studio

其现在已内置支持 Gradle，会自动帮忙调用Gradle去编译你的apk

但是其内部过程你是不清楚的，所以最好学学Gradle，搞清楚过程。

AS调试

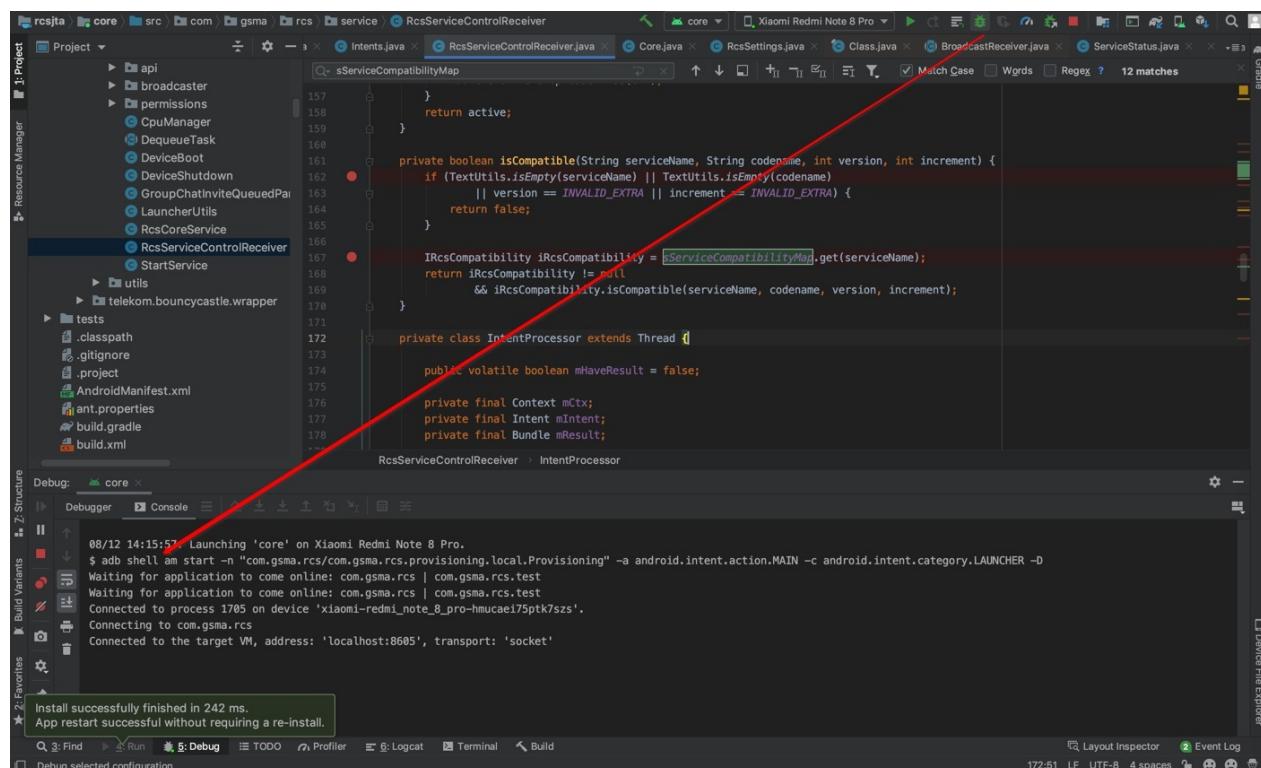
AS调试时就是调用的 adb shell am start

之前就有了解过

```
adb shell am start
```

现在发现，Android Studio去调试时，内部实现也就是调用了：

```
08/12 14:15:57: Launching 'core' on Xiaomi Redmi Note 8 Pro.
$ adb shell am start -n "com.gsma.rcs/com.gsma.rcs.provisioning.local.Provisioning" -a android.intent.action.MAIN -c android.intent.category.LAUNCHER -D
```



AS智能之处

基于IntelliJ IDEA的Android Studio的智能之处

Android Studio 是基于 IntelliJ IDEA 的。

而 IntelliJ IDEA 是 Java 的 IDE , 以智能著称。

下面记录遇到的, 觉得智能的地方

用 Find Usages 去查找被调用地方, 而不要仅仅用搜索

对于

src/com/gsma/rcc/service/StartService.java

的:

```
public class StartService extends Service {
```

想要去找有哪些地方调用了。

结果搜:

StartService(

只找到2处:

The screenshot shows the IntelliJ IDEA interface with the 'Find in Path' dialog open. The search term 'StartService()' is entered in the search bar. The results pane shows two matches found in two files:

- LauncherUtils.java** at line 95: `StartService.LaunchRcsStartService(context, boot, user);`
- StartService.java** at line 584: `static void LaunchRcsStartService(Context context, boolean boot, boolean user) {`

The code editor below shows the implementation of `LaunchRcsStartService` in `LauncherUtils.java`. The method calls `StartService.LaunchRcsStartService`.

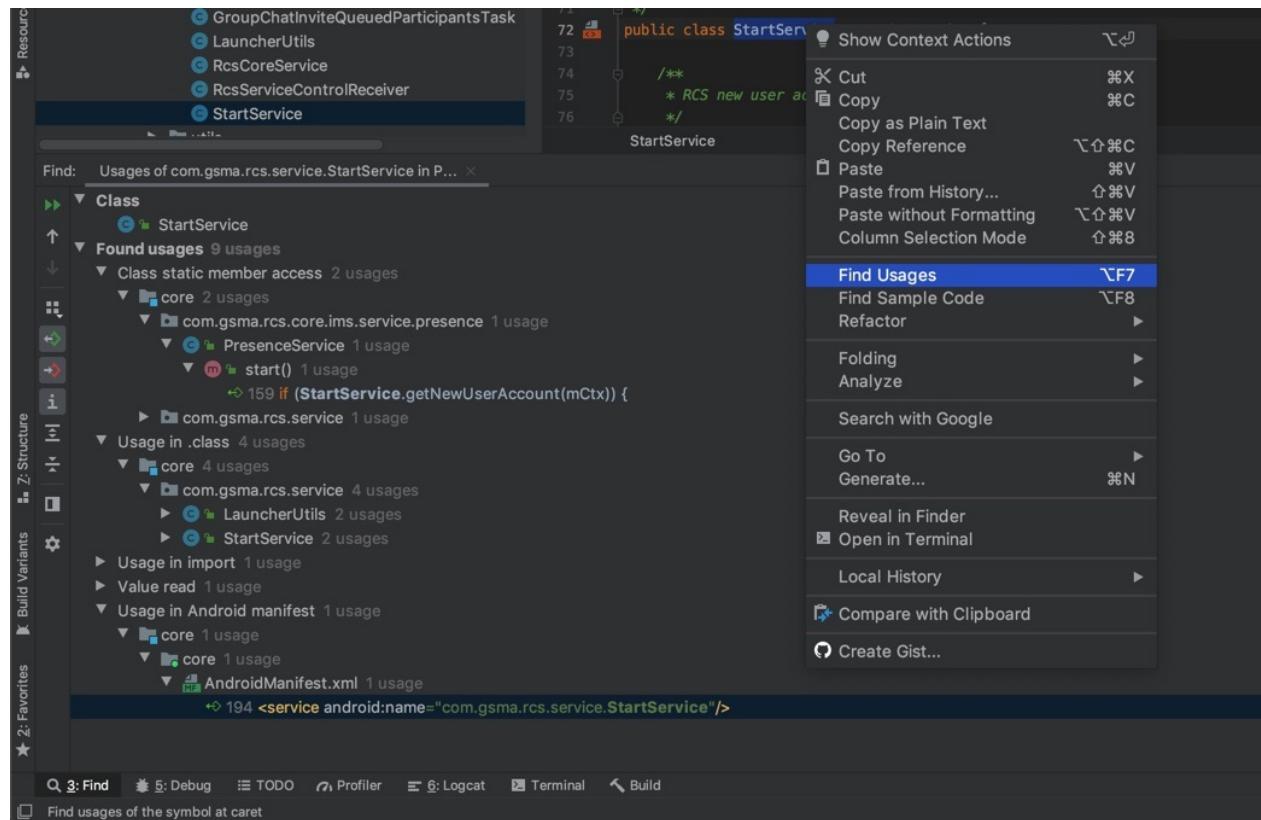
```

89     @ public static void launchRcsService(Context context, boolean boot, boolean user,
90             RcsSettings rcsSettings) {
91             /* Set the logger properties */
92             Logger.sActivationFlag = rcsSettings.isTraceActivated();
93             Logger.traceLevel = rcsSettings.getTraceLevel();
94             if (rcsSettings.isServiceActivated()) {
95                 StartService.LaunchRcsStartService(context, boot, user);
96             }
97         }
98     }
99 /**
100 * Launch the RCS core service
101 *
102 * @param context Application context
103 * @param rcsSettings RCS settings accessor

```

其实是没有真正找到被调用的地方。

而后来想起来了，换用： Find Usages



就可以找到，真正被调用的多处地方。

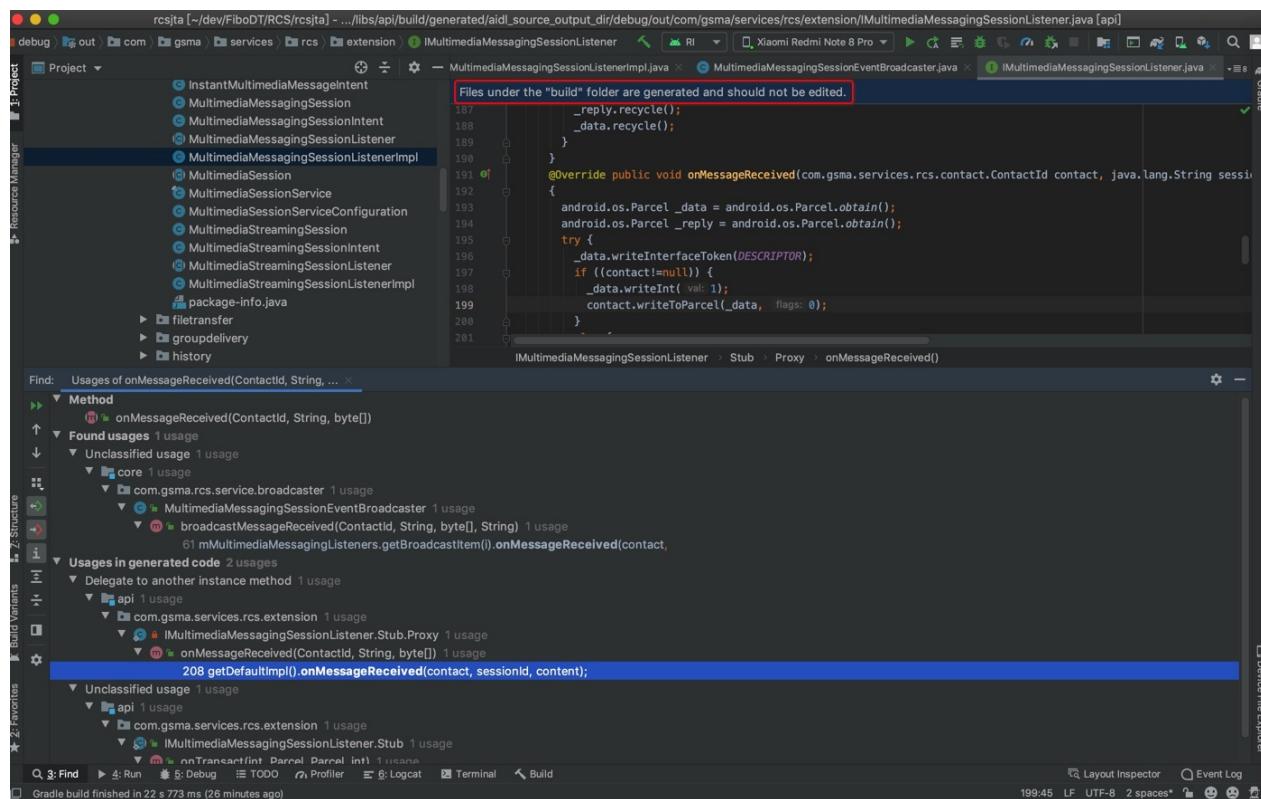
是效率更高，更准确的。

即Find Usages还是很准很好用的。

自动识别build目录下的文件

打开一个java文件，自动提示：

```
Files under the build folder are generated and should not be edited
```



此处文件路径是：

`build/generated/aidl_source_output_dir/debug/out/com/gsma/services/rcs/extension/IMultimediaMessagingSessionListener.java`

即：

在 `build` 目录下的文件，是（项目源码编译后）生成的，不应该被编辑

即：提示你，不要在这里编译，而应该去找源码编辑

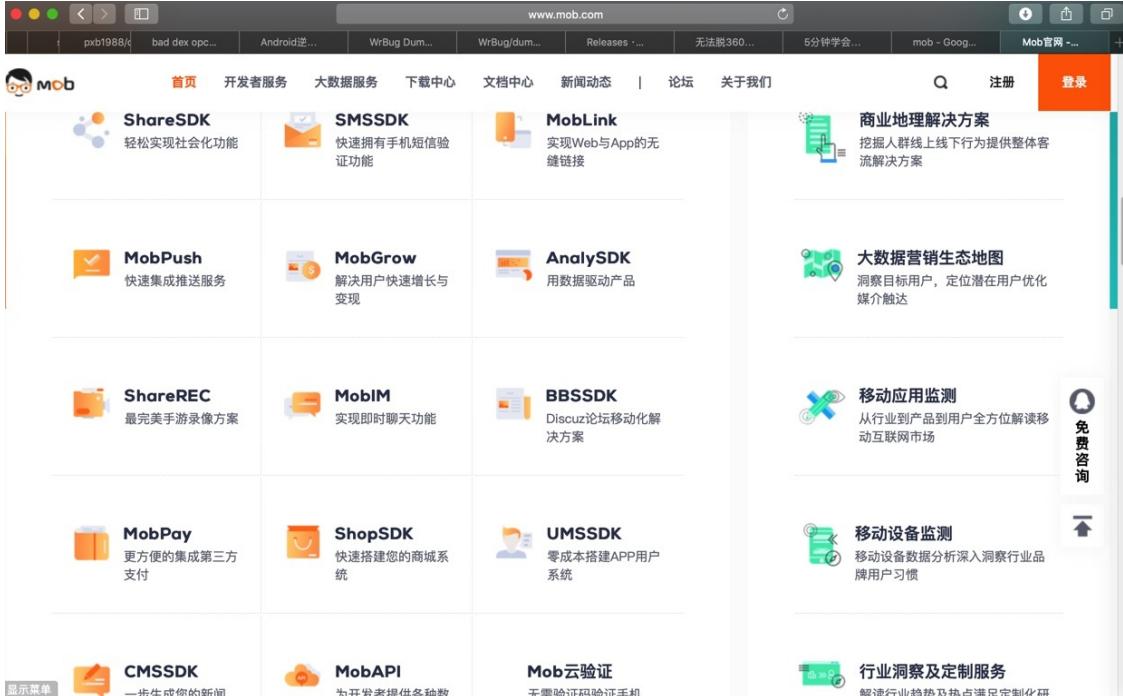
很是智能，防止你不小心浪费了时间在编译生成的地方修改代码。

crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新：2022-11-03 21:44:00

Android常用库

Android中常用的一些库：

- LitePal
 - 作用：方便操作SQLite
 - 仓库
 - <https://github.com/LitePalFramework/LitePal>
 - LitePalFramework/LitePal: An Android library that makes developers use SQLite database extremely easy.
 - 举例：
 - 小花生的安卓app中用LitePal操作SQLite
- anko
 - 库
 - Kotlin/anko: Pleasant Android application development
 - <https://github.com/Kotlin/anko>
 - 资料
 - [使用 Kotlin 和 Anko 的安卓开发入门](#)
 - 作用：方便安卓布局，无需用繁琐的xml去布局
 - 类似于iOS的
 - swift中的Cartography
 - <https://github.com/robb/Cartography>
 - robb/Cartography: A declarative Auto Layout DSL for Swift
 - 或其他的比如：PureLayout
 - <https://github.com/PureLayout/PureLayout>
 - PureLayout/PureLayout: The ultimate API for iOS & OS X Auto Layout — impressively simple, immensely powerful. Objective-C and Swift compatible.
 - 举例
 - 小花生安卓版
 - okio
 - 仓库
 - <https://github.com/square/okio>
 - square/okio: A modern I/O API for Java
 - 作用：方便数据操作
 - 举例
 - 小花生安卓版
 - 其中用到的okhttp依赖于okio
 - okhttp
 - okhttp3
 - 作用：http网络库
 - 仓库
 - <https://github.com/square/okhttp>
 - square/okhttp: An HTTP+HTTP/2 client for Android and Java applications.
 - 举例：
 - 趣配音安卓版
 - 小花生安卓版
 - afinal
 - 作用：Afinal 是一个android的sqlite orm 和 ioc 框架。同时封装了android中的http框架，使其更加简单易用
 - 仓库
 - [yangfuhai/afinal](https://github.com/yangfuhai/afinal): Afinal是一个android的ioc, orm框架：
 - 方便通过注解的方式进行绑定ui和事件

- 方便的加载bitmap图片
 - 方便操作SQLite
 - 方便以ajax形式请求http数据
 - <https://github.com/yangfuhai/afinal>
- 举例:
 - 小花生安卓版
- PermissionGen
 - 作用：方便申请权限
 - 背景：android在6.0系统以后，权限申请变得麻烦起来，所以用库去方便权限申请
 - 代码仓库
 - <https://github.com/lovedise/PermissionGen>
 - lovedise/PermissionGen: Android API easy to use permission for Android M
 - 举例:
 - 小花生安卓版
- mob
 - 功能：
 - 社交分享、短信验证、深度链接、统计分析、推送等
 - Mob依托开发者服务，构建全景数据生态，挖掘大数据的价值
 - 
 - 比如：
 - ShareSDK
 - 仓库：
 - <https://github.com/MobClub/ShareSDK-for-Android>
 - smssdk
 - 主页：<http://www.mob.com>
 - loopj
 - 作用：异步http库
 - 仓库
 - <https://github.com/loopj/android-async-http>
 - loopj/android-async-http: An Asynchronous HTTP Library for Android
 - 举例
 - 小花生安卓版
 - xutil
 - 作用：http网络库
 - 仓库：
 - 小花生安卓版

- <https://github.com/wyouflf/xUtils3>
 - wyouflf/xUtils3: android orm, bitmap, http, view inject...
- 举例:
 - 小花生安卓版
 - [已解决] xutils3如何实现http的put请求request
- gson
 - 作用: 处理json
 - 仓库:
 - <https://github.com/google/gson>
 - google/gson: A Java serialization/deserialization library to convert Java Objects into JSON and back
 - 举例:
 - 小花生安卓版
- zxing
 - 作用: 处理二维码
 - 仓库:
 - <https://github.com/zxing/zxing>
 - zxing/zxing: ZXing ("Zebra Crossing") barcode scanning library for Java, Android
 - 举例:
 - 小花生安卓版
- subsampling-scale-image-view
 - 作用: 图片缩放处理显示
 - 仓库:
 - <https://github.com/davemorrissey/subsampling-scale-image-view>
 - davemorrissey/subsampling-scale-image-view: Android library (AAR). Highly configurable, easily extendable deep zoom view for displaying huge images without loss of detail. Perfect for photo galleries, maps, building plans etc.
 - 举例:
 - 小花生安卓版
- PhotoPick-Master
 - 作用: 图片选择
 - 仓库:
 - <https://github.com/Awent/PhotoPick-Master>
 - Awent/PhotoPick-Master: A photo pick, clip, view larger image tool. and you can define some parameter
- android-pickers
 - 作用: 选择器
 - 仓库:
 - <https://github.com/addappcn/android-pickers>
 - addappcn/android-pickers: 安卓选择器类库

Android常用代码段

crifanAndroid.py

在折腾移动端自动化：

安卓自动化测试利器：uiautomator2

期间，使用 Python 代码，对于 Android 设备的操作，整理了一些常用的代码逻辑，代码段，代码块。

已整理并发布到自己Crifan的库中了：

- <https://github.com/crifan/crifanLibPython/blob/master/python3/crifanLib/thirdParty/crifanAndroid.py>

截至 20221003，最新代码是：

```
# usr bin python
# coding utf 8
"""

Filename crifanAndroid.py
Function crifanLib's Android related functions
Version 20210114
Latest: https://github.com/crifan/crifanLibPython/blob/master/python3/crifanLib/thirdParty/crifanAndroid.py
"""

__author__ = "Crifan Li (admin@crifan.com)"
__version__ = "20210114"
__copyright__ = "Copyright (c) 2021, Crifan Li"
__license__ = "GPL"

import re
import logging

from crifanLib crifanFile import getCommandOutput

#####
# Config
#####

#####
# Constant
#####
CURRENT_LIB_FILENAME = "crifanAndroid"

#####
# Global Variable
#####

#####
# Internal Function
#####

#####
# Local Function
#####

def getAndroidDeviceList(self, isGetDetail=False):
    """Get android device list

    Args
        isGetDetail (bool): True to use `adb devices -l`, False to use `adb devices`
    Returns
        device list(list)
    Raises
    Examples
        output
            False -> ['2e2a0cb1', 'orga4pmzee4ts47t', '192.168.31.84:5555']
            True -> [{"2e2a0cb1": {"usb": "338882102X", "product": "PD2065", "model": "V2065A", "device": "PD2065", "transport_id": "4"}}, {"orga4pmzee4ts47t": {"usb": "338886656X", "product": "atom", "model": "M2004J7AC", "device": "M2004J7AC"}]
    """
    pass
```

```

        atom', 'transport_id': '24')}, {'192.168.31.84:5555': {'product': 'PD2065', 'model': 'V2065A', 'device': 'PD2065', 'transport_id': '5'}}]
    """
    deviceList = []

    getDevicesCmd = 'adb devices'
    if isGetDetail:
        getDevicesCmd += " -l"
    logging.debug("getDevicesCmd=%s", getDevicesCmd)

    isRunOk, deviceLines = getCommandOutput(getDevicesCmd)
    logging.debug("isRunOk=%s, deviceLines=%s", isRunOk, deviceLines)
    # ['List of devices attached', '2e2a0cb1\tdevice', 'orga4pmzee4ts47t\tdevice', '192.168.31.84:5555\tdevice', '']
    if not isRunOk:
        return deviceList

    """
    adb devices :
        List of devices attached
        2e2a0cb1    device
        orga4pmzee4ts47t    device
        192.168.31.84:5555      device
    """

    """
    adb devices -l:
        List of devices attached
        2e2a0cb1          device usb 338952192X product PD2065 model V2065A device PD2065 transport_id 4
        orga4pmzee4ts47t    device usb 338886656X product atom model M2004J7AC device atom transport_id 24
        192.168.31.84:5555    device product PD2065 model V2065A device PD2065 transport_id 5
    """

    for eachLine in deviceLines:
        if not eachLine:
            continue

        if "devices attached" in eachLine:
            continue

        foundDevice = re.search("(?P<devSerial>[\w.\:]+)\s+device\s*(?P<devDetail>[\w\: ]+)?", eachLine)
        logging.debug("foundDevice=%s", foundDevice)
        # foundDevice= re.Match object; span=(0, 101), match='2e2a0cb1'           device usb 338952192X prod>
        if foundDevice:
            devSerial = foundDevice.group("devSerial")
            logging.debug("devSerial=%s", devSerial)
            # devSerial 2e2a0cb1
            if isGetDetail:
                devDetail = foundDevice.group("devDetail")
                logging.debug("devDetail=%s", devDetail)
                # devDetail=usb 338952192X product PD2065 model V2065A device PD2065 transport_id 4
                keyValueIter = re.finditer("(?P<key>\w+):(?P<value>\w+)", devDetail) # callable_iterator object at 0
                x10baa3a00>
                keyValueMatchList = list(keyValueIter)
                logging.debug("keyValueMatchList=%s", keyValueMatchList)
                # keyValueMatchList [ re.Match object; span=(0, 14), match='usb:338952192X', <re.Match object; span=(15, 29), match='product:PD2065', <re.Match object; span=(30, 42), match='model:V2065A', <re.Match object; span=(43, 56), match='device:PD2065', <re.Match object; span=(57, 71), match='transport_id:4' ]
                detailInfoDict = {}
                for eachMatch in keyValueMatchList:
                    eachKey = eachMatch.group("key")
                    eachValue = eachMatch.group("value")
                    detailInfoDict[eachKey] = eachValue
                logging.debug("detailInfoDict=%s", detailInfoDict)
                # detailInfoDict {'usb': '338952192X', 'product': 'PD2065', 'model': 'V2065A', 'device': 'PD2065', 'transport_id': '4'}
                curDevDetailDict = {
                    devSerial: detailInfoDict
                }
                logging.debug("curDevDetailDict=%s", curDevDetailDict)
                # curDevDetailDict={'2e2a0cb1': {'usb': '338952192X', 'product': 'PD2065', 'model': 'V2065A', 'device': 'PD2065', 'transport_id': '4'}}
                deviceList.append(curDevDetailDict)
            else:
                deviceList.append(devSerial)

    logging.info("deviceList=%s", deviceList)
    # deviceList [{('2e2a0cb1': {'usb': '338952192X', 'product': 'PD2065', 'model': 'V2065A', 'device': 'PD2065', 'transport_id': '4'})}, {('orga4pmzee4ts47t': {'usb': '338886656X', 'product': 'atom', 'model': 'M2004J7AC', 'device': 'ato

```

```

m', 'transport_id': '24'}}, {'192.168.31.84:5555': {'product': 'PD2065', 'model': 'V2065A', 'device': 'PD2065', 'transport_id': '5'}}]
# ['2e2a0ccb1', 'orga4pmzee4ts47t', '192.168.31.84:5555']
return deviceList

def isAndroidUsbConnected(self, deviceSerialId):
    """Check whether android device is currently USB wired connected or not

    Args
        deviceSerialId (str): android devivce serial id
    Returns
        connected or not (bool)
    Raises
    Examples
        input: "orga4pmzee4ts47t"
        output: True
    """
    isUsbConnected = False
    isRealSerialId = re.search("\w+", deviceSerialId)
    if not isRealSerialId:
        # makesure is not wifi, such as: 192.168.31.84:5555
        logging.error("Invalid android USB wired connected device serial id %s", deviceSerialId)
        return isUsbConnected

    deviceDetailList = self.getAndroidDeviceList(isGetDetail=True)
    for eachDevDetailDict in deviceDetailList:
        curDevSerialStr, curDevDetailDict = list(eachDevDetailDict.items())[0]
        if deviceSerialId == curDevSerialStr:
            detailInfoKeyList = list(curDevDetailDict.keys())
            # ['usb', 'product', 'model', 'device', 'transport_id']
            if "usb" in detailInfoKeyList:
                isUsbConnected = True
            break

    return isUsbConnected

def androidConnectWiFiDevice(self, wifiSerial):
    """Use Android 'adb connect' to connect WiFi wireless devive

    Args
        wifiSerial (str): android devivce WiFi serial, eg: 192.168.31.84:5555
    Returns
        connect ok or not (bool)
    Raises
    Examples
        input: "192.168.31.84:5555"
        output: True
    """
    isConnectOk = False

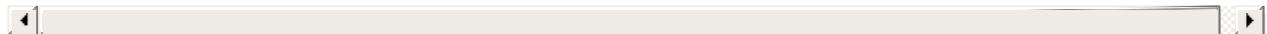
    adbConnectCmd = "adb connect %s" % wifiSerial
    logging.info("Try connect Android device: %s", adbConnectCmd)
    # os.system(adbConnectCmd) # when failed, will wait too long time: ~ 1 minutes
    isRunOk, cmdOutputStr = getCommandOutput(adbConnectCmd, timeout=1)
    logging.info("isRunOk=%s, console output: %s", isRunOk, cmdOutputStr)
    # connected to 192.168.31.84:5555
    # already connected to 192.168.31.84:5555
    # failed to connect to '192.168.31.84:5555': Operation timed out
    # "failed to connect to '192.168.31.84:5555': Connection refused\n"
    # err=Command 'adb connect 192.168.31.84:5555' timed out after 1 seconds when run cmd adb connect 192.168.31.84:5
    if cmdOutputStr:
        if "connected" in cmdOutputStr:
            isConnectOk = True
        elif ("failed" in cmdOutputStr) or ("timed out" in cmdOutputStr):
            isConnectOk = False
    else:
        isConnectOk = False

    return isConnectOk

#####
# Test
#####

if __name__ == '__main__':
    print("[oricanLib-%s] %s" % (CURRENT_LIB_FILENAME, __version__))

```



crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook 最后更新: 2022-11-03 17:54:41

附录

下面列出相关参考资料。

crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新: 2022-03-17 20:39:28

参考资料

- 【未解决】Android项目rcsjta中如何才能运行到rcs的core的service
- 【整理】Android的sdk数值和版本号对应关系映射表
- 【整理】Android的api和sdk的版本号安卓版本和名称对应关系
- 【已解决】Android相关环境变量定义：ANDROID_SDK_ROOT和ANDROID_HOME
- 【已解决】Android相关环境初始化build-tools和ANDROID_SDK_ROOT
- 【已解决】Android环境初始化：相关环境变量
- 【已解决】Mac中安装android Studio和android SDK
- 【已解决】Mac M2 Max中初始化和安装Android Studio
-
- [支付宝行走支持的机型_红米手机1_MIUI论坛](#)
- [How do I navigate to the start or end of a file in Android Studio? - Stack Overflow](#)
- [Gradle 完整指南（Android） - Android - 掘金](#)
- [SDK Platform 版本说明 | Android 开发者 | Android Developers](#)
- [代号、标记和细分版本号 | Android 开源项目 | Android Open Source Project](#)
- [Android常用开源框架 | 笔沫拾光的博客](#)
- [环境变量 | Android 开发者 | Android Developers](#)
-

crifan.org, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved, powered by Gitbook最后更新：2023-07-24 22:01:36