

# Windows下Android打包教程：将Android源码打包成APK可执行文件

原创

飞翔的大黑壮(猫猫) 于 2020-01-20 16:41:52 发布 5441 收藏 11

分类专栏：[高级测试](#) [测试基础部分](#) [手机专项测试](#)

只做原创，告别转载。

本文链接：[https://blog.csdn.net/pingsha\\_luoyan/article/details/104051411](https://blog.csdn.net/pingsha_luoyan/article/details/104051411)

版权



[高级测试](#) 同时被 3 个专栏收录

73 篇文章 3 订阅

订阅专栏



[测试基础部分](#)

25 篇文章 3 订阅

订阅专栏



[手机专项测试](#)

5 篇文章 0 订阅

订阅专栏

## 最基出的环境：

Node.js, Android SDK, Java JDK(1.8)

nodejs安装包中文下载地址：<http://nodejs.cn/download/>

Android SDK下载地址：<https://www.androiddevtools.cn/>

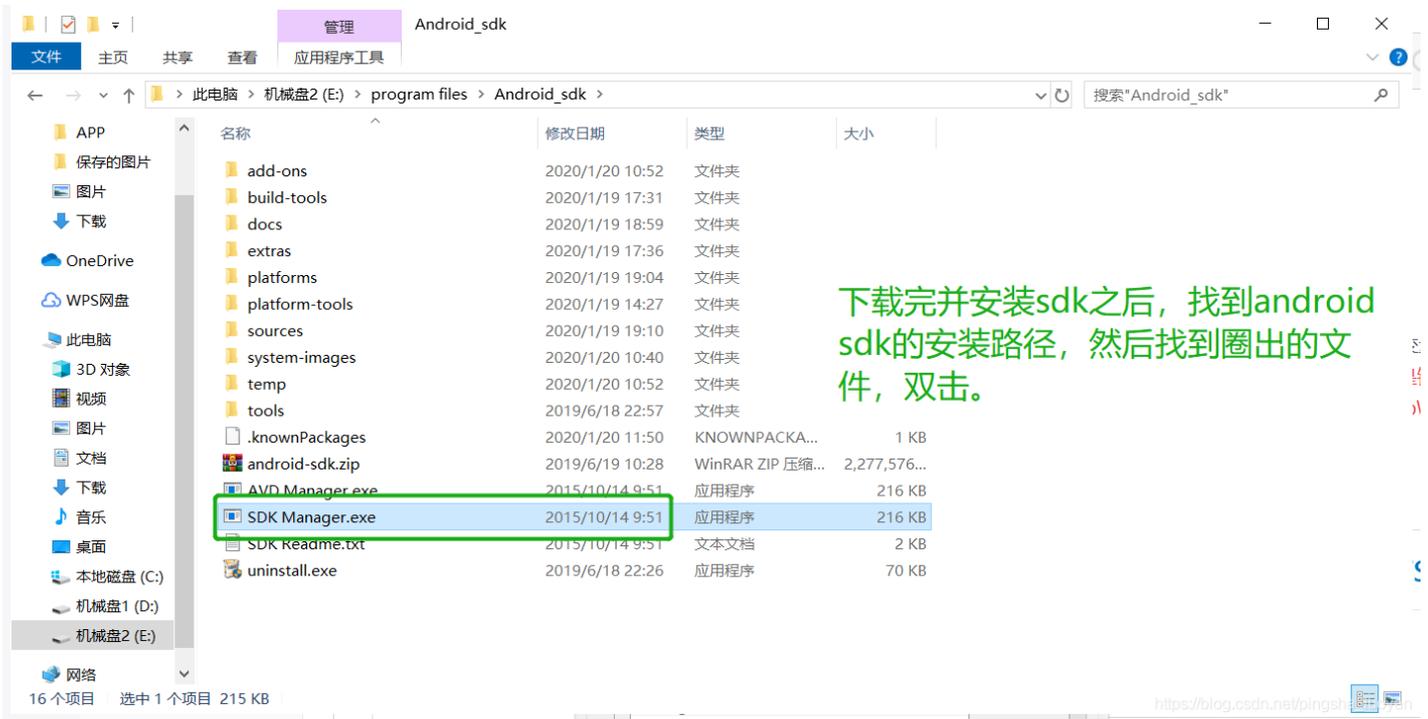
Java JDK下载地址：

## 打包环境搭建：

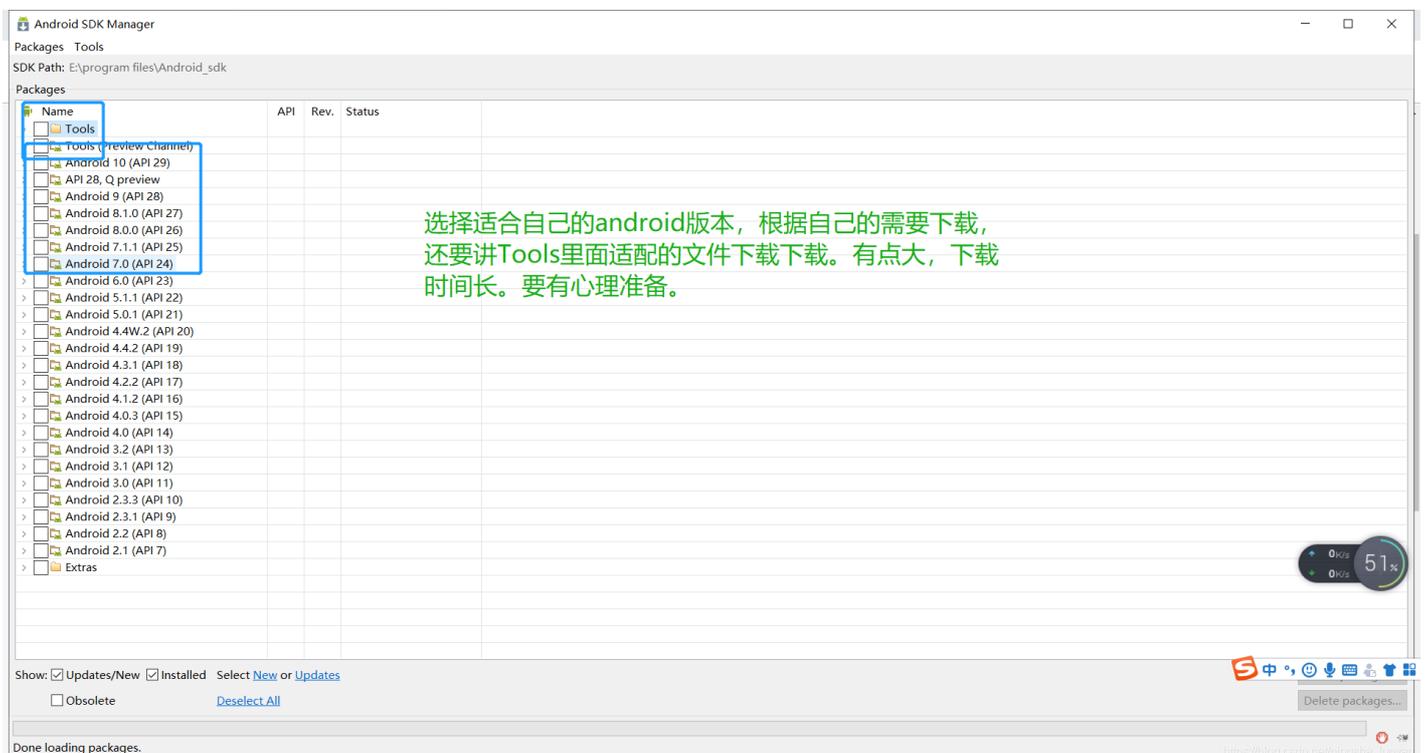
1. 首先安装Java JDK，配置Java JDK环境变量。

为什么安装java: android sdk是依赖java的，只有安装了java环境，才能保证android sdk的正常运行

2. 安装Android SDK，配置Android SDK的环境变量。



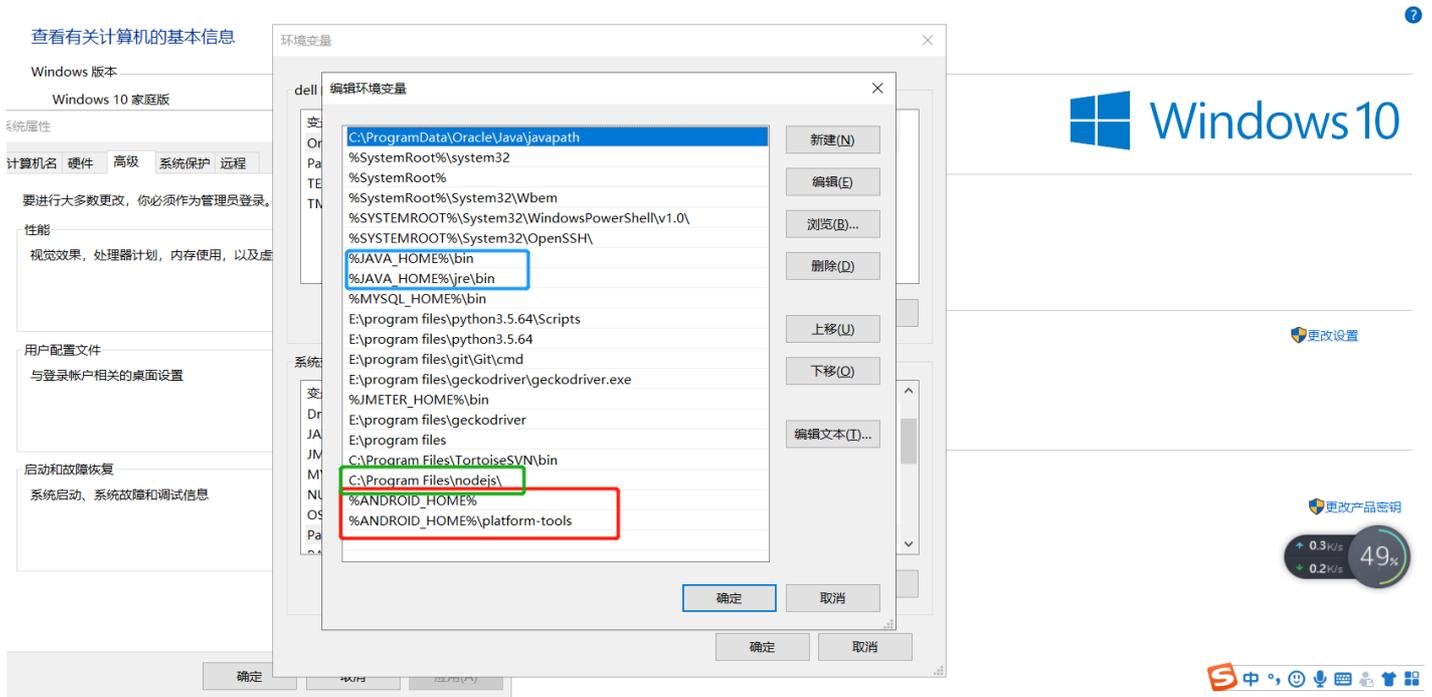
下载完并安装sdk之后，找到android sdk的安装路径，然后找到圈出的文件，双击。



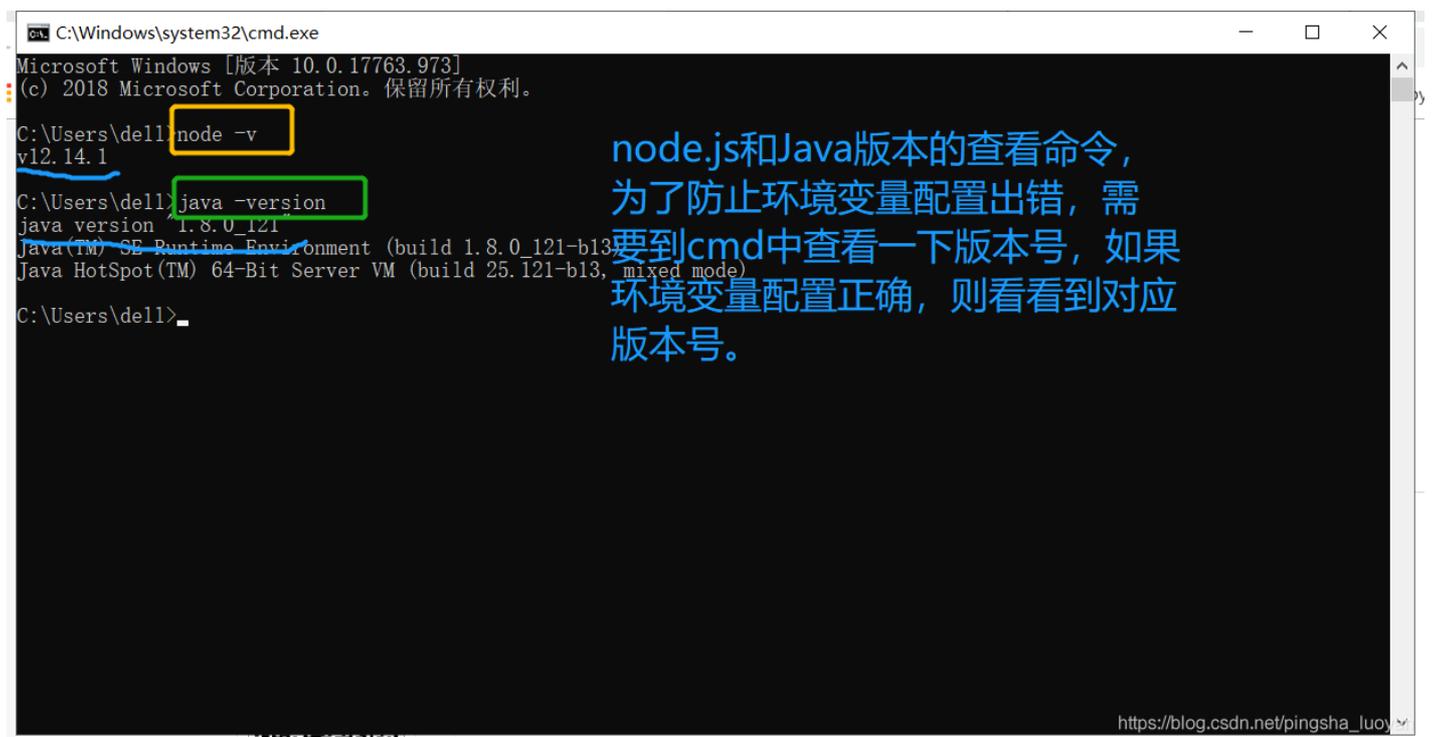
选择适合自己的android版本，根据自己的需要下载，还要讲Tools里面适配的文件下载下载。有点大，下载时间长。要有心理准备。

3.安装Node.js, Node.js在安装时，自动配置好环境变量，但是存在没有配置的情况，需要自己确认一下。

例如：这是我自己电脑上的配置环境，环境变量path下必须有java，node.js，Android的环境变量，为了防止环境变量路径输入错误，需要进入到cmd中查看，不要嫌弃繁琐，认为确认是没有必要的，但是请明白，环境如果搭建不好，一切都是镜花水月。（现身说法：刚开始学习java的时候，那时候使用的windo 7系统，配置环境变量时，拉掉了一个;导致最基出的HelloWorld都实现不了。）



[https://blog.csdn.net/pingsha\\_luoyan](https://blog.csdn.net/pingsha_luoyan)



基础环境已经搭建好。你认为这样就结束了吗，然而并没有，

```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.17763.973]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\dell>node -v
v12.14.1

C:\Users\dell>java -version
java version "1.8.0_121"
Java(TM) SE Runtime Environment (build 1.8.0_121-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.121-b13, mixed mode)

C:\Users\dell>npm -v
6.13.4

C:\Users\dell>
```

npm在node.js安装后就自动加载了，同时也需要自己确认一下。



[https://blog.csdn.net/pingsha\\_luoyan](https://blog.csdn.net/pingsha_luoyan)

下面是硬菜了：

说明：打包分为两种，第一张密钥打包，另外一种非密钥打包，本次介绍的是密钥打包，其实密钥打包和非密钥打包基本上相似，知识密钥打包多了几个小步骤而已。

1.没有密钥生成密钥，若是已经有密钥，则将密钥放在android\app目录下。

生成密钥的命令：

```
$ keytool -genkeypair -v -keystore my-release-key.keystore -alias my-key-alias -keyalg RSA -keysize 2048 -v
```

解释密钥生成命令：

1.在 Windows ， keytool命令在Java JDK 的 bin 目录中。

2.alias后面跟着的是密钥的别名，my-key-alias，可以是任意字符，只要不提示错误即可。因一个证书库中可以存放多个证书，通过别名标识证书

3.keyalg后跟的是加密的算法：RSA ， 密钥算法有三种：RSA、DSA、EC

4.keysize 后跟的是密钥长度，2048。（keyalg要注意与keysize 相对应）

RSA、DSA、EC 分别对应 2048、1024、256

5.validity :证书保质期，在多少天内有效

2.设置gradle环境：

将生成的密钥放置到android\app目录下，之后修改android\gradle.properties文件。

```
MYAPP_RELEASE_STORE_FILE=my-release-key.keystore
MYAPP_RELEASE_KEY_ALIAS=my-key-alias
MYAPP_RELEASE_STORE_PASSWORD=*****
MYAPP_RELEASE_KEY_PASSWORD=*****
```

3.将签名配置加到gradle配置中。

android/app/build.gradle中添加。

```
android {
    ...
    defaultConfig { ... }
    signingConfigs {
        release {
            if (project.hasProperty('MYAPP_RELEASE_STORE_FILE')) {
                storeFile file(MYAPP_RELEASE_STORE_FILE)
                storePassword MYAPP_RELEASE_STORE_PASSWORD
                keyAlias MYAPP_RELEASE_KEY_ALIAS
                keyPassword MYAPP_RELEASE_KEY_PASSWORD
            }
        }
    }
    buildTypes {
        release {
            ...
            signingConfig signingConfigs.release
        }
    }
}
```

4.生成apk包。

进入android目录下，执行：

```
gradlew assembleRelease
```

成功后就可以在目录android/app/build/outputs/apk/release/下找到app-release.apk包了。