## Android 各个SdkVersion的含义

```
C ExampleUnitTest.java
                                                                   local.properties
                                                                                         helloworld2020_0312.iml
    app app
                settings.gradle
             apply plugin: 'com.android.application'
ect
    1
    3
             android {
                  compileSdkVersion 29
                  buildToolsVersion "29.0.2"
                  defaultConfig {
    6
                       applicationId "com.example.helloworld"
    8
                       minSdkVersion 25
    9
                       targetSdkVersion 29
   10
                       versionCode 1
   11
                       versionName "1.0"
                       testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
   12
   13
                  buildTypes {
   14
   15
                       release {
                           minifyEnabled false
   16
   17
                            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-
   18
                  }
   19
             } 🕊
   20
   21
   22
             dependencies {
   23
                  implementation fileTree(dir: 'libs', include: ['*.jar'])
                  implementation 'androidx.appcompat:appcompat:1.0.2'
implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
   24
   25
   26
                  testImplementation 'junit:junit:4.12'
                  androidTestImplementation androidTestImplementation androidTestImplementation androidTestImplementation androidX.test.espresso:espresso-core:3.1.1
   27
   28
   29
   30
```

在使用AndroidStudio的时候,build.gradle中要配置好多的SDK的版本号,有时候搞的我们眼花缭乱,现在就说说这几个版本的含义

**compileSdkVersion**: 只编译工程的SDK版本,我们知道Android Studio中的内置的Gradle来编译工程的,就是告诉Gradle我要使用哪个版本的SDK来编译工程。

**buildToolsVersion:**是你构建工具的版本,其中包括了打包工具aapt、dx等等。这个工具的目录位于..your\_sdk\_path/build-tools/XX.XX.XX。这个版本号的选择一般和上面的编译版本号的大版本号是一致的,仔细观察两个版本号,上面的编译版本号只有大版本号,而这个构建工具的版本是有小版本号的。

可以用高版本的build-tool去构建一个低版本的sdk工程,例如build-tool的版本为20,去构建一个sdk版本为18的

minSdkVersion: 这个就是项目能够在最小多少版本上的手机上运行,也就指定了你最低能用的API的范围

**targetSdkVersion:** 目标SDK版本,因为Android版本是向前兼容的,就是新发布的 SDK版本是兼容老的SDK版本的,所以这个版本号的设置就是提供向前兼容的主要方式。这个版本号意思是: 你在安装了这个版本的手机上已经做的充分的测试了。

我知道你读到这里肯定对上面的这句话有疑问「这个版本号的设置就是提供向前兼容的主要方式」,这句话到底是什么意思呢?就是说随着Android版本的不断升级,某一个API在不同的版本上会发生变化,但是老的apk都是基于老的API来开发的,为了保证老的apk也能在安装了新系统的手机上运行所以出现了这个targetSdkVersion,当你的apk是在设置成22的时候开发的,即时你的apk安装在Android6.0(API23)系统的手机上代码运行调用的API也都是 22 上的,系统是怎么做的呢?在源码总是通过获取到你设置的 targetSdkVersion,然后if——else执行的不同的逻辑。

一般compileSdkVersion和targetSdkVersion这两个版本号使用最新的版本比较好,这两个一般设置成一样的。

上面4个概念,前两个主要是编译时使用的,后两个主要控制的是运行的时候手机的 版本

但是实际开发中如果你有这样的需求呢?你需要用新发布系统的API,但是你的apk中使用的API在新系统中变化了,你不想改动那些变化的API,怎么办呢?你可能已经想到了,就是把compileSDKVersion版本改成高版本,targetSdkVersion还是原来的版本,这样就可以了。但是在使用新API的代码中会红色警告,警告如下

Call requires API level 23 (current min is 19); android.telephony.TelephonyManager#getDeviceId more... (Ctrl+F1)

经过测试,虽然这样警告,但是可以编译运行,而且API的调用正常。可以放心使用。