Android Studio入门到精通

AS简介

经过2年时间的研发，Google终于正式发布了面向Android开发者的集成开发环境Android Studio 1.2（稳定版）。Android Studio是Google开发的一款面向Android开发者的IDE，支持Windows、Mac、Linux等操作系统，基于流行的Java语言集成开发环境IntelliJ搭建而成。该IDE在2013年5月的Google I/O开发者大会上首次露面，当时的测试版各种莫名其妙的Bug，但是14年12月8日发布的版本是稳定版。Android Studio 1.0推出后，Google官方将逐步放弃对原来主要的Eclipse ADT的支持，并为Eclipse用户提供了工程迁移的解决办法。不过相信作为Developer的你上手AS 1.0以后你再也不愿意使用原来苦逼的Eclipse+ADT了，你会被AS的各种强大所吸引。

下载安装

下载AS前先说下，AS安装包分为含SDK版本和不含SDK版本下载，如果你有SDK，那么完全可以下载不含SDK版本；不过下载了含SDK版本也没事，安装时选择自定义SDK也可以，安装后重新指定SDK路径也可以，总之看个人爱好喽。先吐槽下天朝的强大吧，不得不拜服天朝的墙。如果你有梯子请去Android Developer下载最新版的AS安装包，如果你没有梯子那也有个办法，就是去Android Studio中文社区官网下载你的平台需要的安装包。

下载下来以后安装的过程可以忽略了吧，能安装的都是程序猿吧，所以安装这点就不说了，注意已经正确安装配置了JDK。

安装好了以后首次运行AS可能一直停在Fetching Android SDK component information。如下界面：

这是因为天朝的墙真的太高太厚把首次运行更新SDK给墙了。解决办法就是关闭安装向导，如果无法关闭可以在任务管理器中手动关掉进程（Ctrl+Alt+Del启动任务管理器），然后打开AS安装目录下的bin目录里面的idea.properties文件，添加一条禁用开始运行向导的配置项：

disable.android.first.run=true

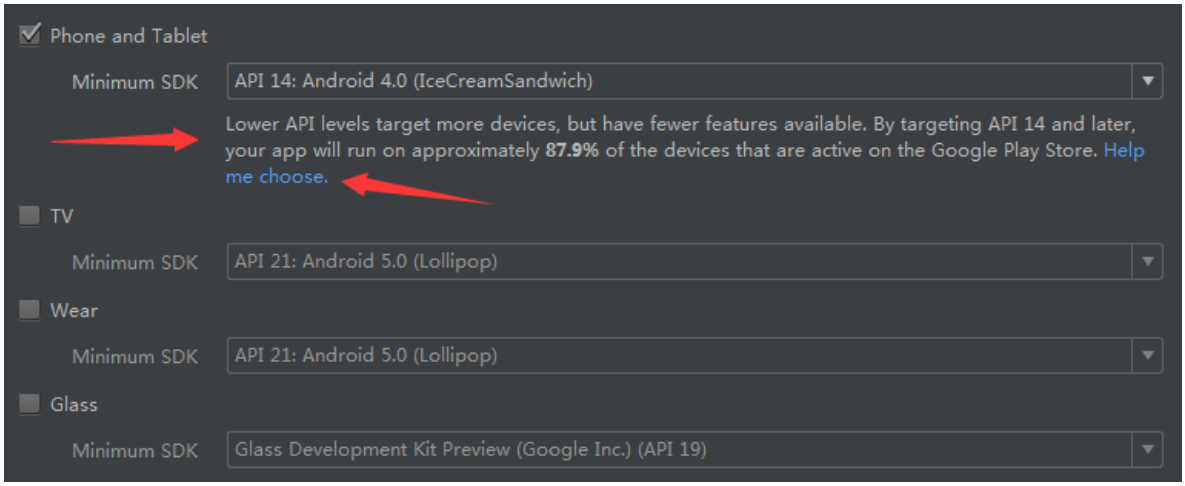
然后再启动程序就会打开项目向导界面，这个时候如果点击Start a new Android Studio project是没有反应的，并且在Configure下面的SDK Manager是灰色的，这是因为没有安装Android SDK的缘故。这时候一般有两种做法：

1. 自己没有SDK，需要从网络下载；打开向导的Configure-Settings，在查找框里面输入proxy，找到下面的HTTP Proxy，设置代理服务器，并且将Force https://… sources to be fetched using http://选中，然后退出将上面在idea.properties配置文件中添加的那条配置项注释掉重新打开Android Studio等刚开始的向导把Android SDK下载安装完成就可以了。
2. 自己有SDK，重新指定SDK路径；打开向导的Configure->Project Defaults->Project Structure，在此填入你已有的SDK路径。

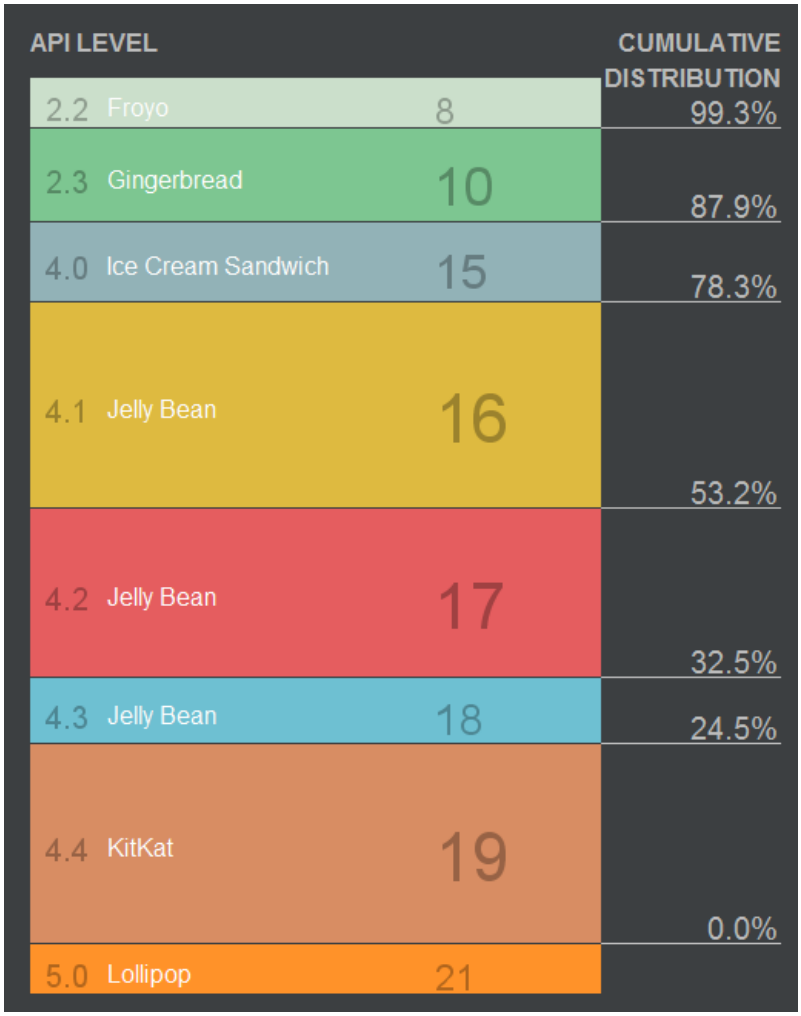
此时重启AS就可以在向导里新建Android工程喽。至此整个安装过程结束。

基本使用介绍

首先新建工程，输入工程名和主包名和存储路径；点击next到如图步骤：



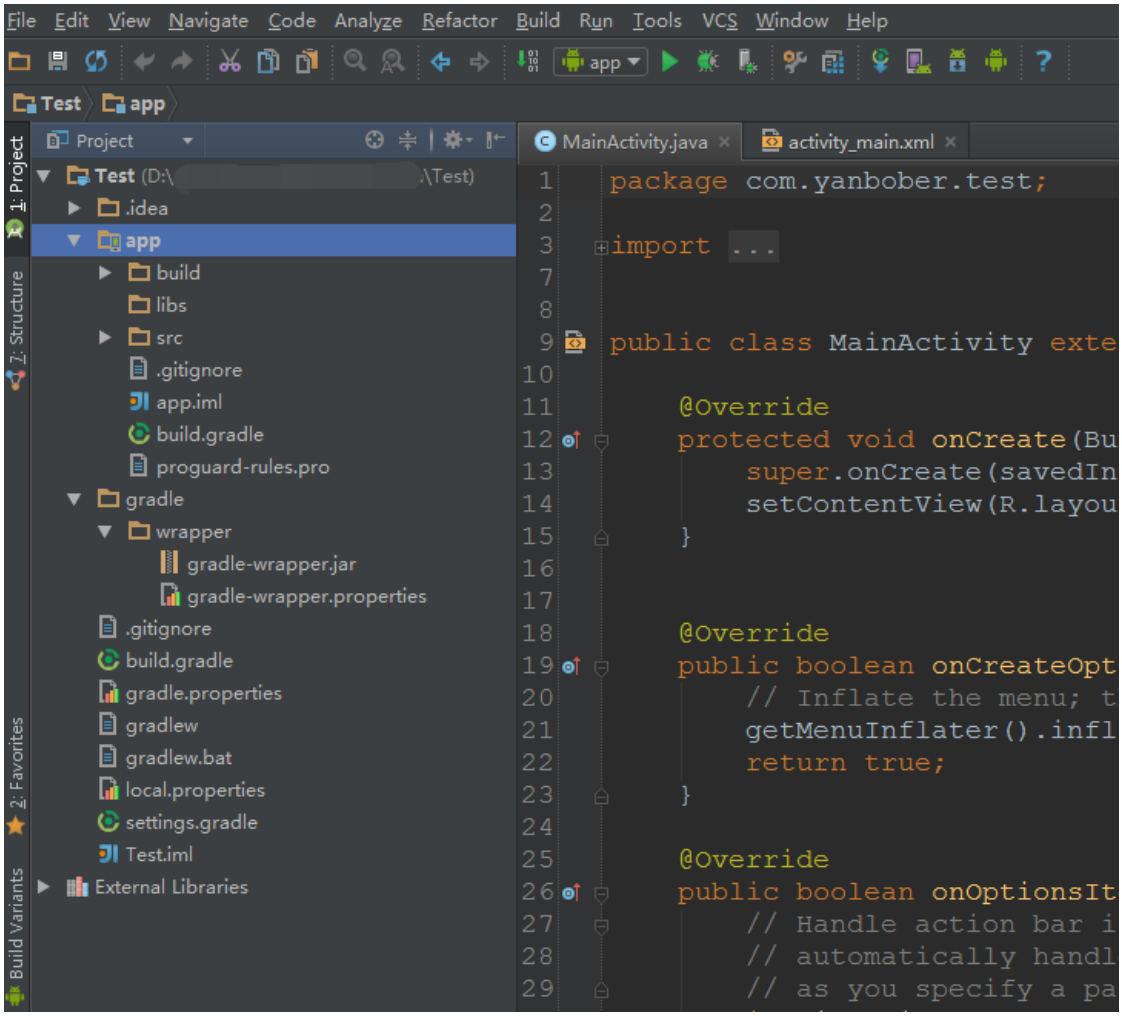
上图中首先你可选择你的App要适配的设备是Wear还是Mobile还是TV。在你新建App选择最低适配版本时，强大的AS会给你一些有用的统计提示，如图描述了当前版本的用户情况，点击Help me choose后弹出如下更加形象的分布图表描述：



爱不释手的亮点就是这么一步一步比Eclipse强大的，这只是一些不值得一提的小点而已，强大的功能还在后面。继续点击Next选择形象友好的GUI模板，点击完成进入工程初始化过程。

第一次安装工程初始化时由于需要联网下载gradle会比较慢，不过有时候不是第一也会慢，工程依赖的gradle版本不匹配时也会自动重新下载；我的初始化很快，原因是我本地的gradle-2.2-all.zip之前已经下载OK的。至于啥时gradle后文会有说明。这儿只是告诉你若果你看到卡一会儿时正常的。

接下来进入到了工程界面下：



这个创建过程可比Eclipse上长的多。主要是因为从gradle上下载。gradle也可以手动离线下载好放在对应目录下。工程的结构和Eclipse上的不同，src下分为java和res。AS是基于idea，而idea和eclipse有大的区别，有好处也有不好的地方，在一段时间里，idea被认为是开发java最好用强大的ide工具，所以AS新建的时候有new application和new module开发。idea没有工作空间这样的说法。这就是Eclipse用户切换过来第一个比较不适应的地方。

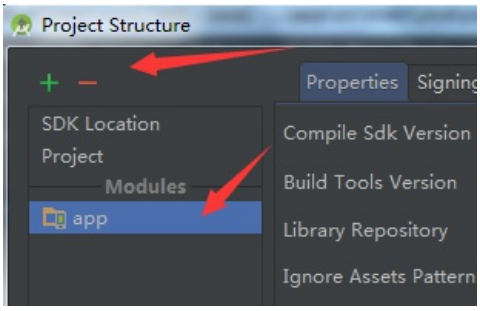
具体说就是：

1. android studio是单工程的开发模式
2. android studio中的application相当于eclipse里的workspace概念
3. android studio中的module相当于eclipse里的project概念

有了如上三条概念自己手动创建摸索下，相信聪明的你自然就明白咋回事了吧。

接下来看一些工欲善其事必先利其器的基本高频率实用设置：

1. 中文乱码——在窗口中，找到IDE Settings->Appearance，在右侧勾选上“Override default fonts by”，然后在第一个下拉框中选择字体为“simsun”，然后apply，重启IDE，就好了。
2. 设置快捷键——在settings窗口中，找到IDE Settings->keymap，右侧打开的就是快捷键了。右键单击要修改的快捷键，会弹出一个菜单，选择“Add keyboard shortcut”就可以修改快捷键了。删除的话，在弹出的菜单中选择remove XXX即可。特别说明，在AS的快捷键设置里可以直接设置使用Eclipse快捷键还是别的IDE快捷键。如果你热衷Eclipse那么也可设置成Eclipse的快捷键。
3. 修改主题——在IDE Settings->Appearance，右侧的Theme选择自己喜欢的主题即可。个人比较喜欢Darcula主题，也就是如上截图样式。
4. 如何将Eclipse工程导入AS使用——选择File->Import Project，在弹出的菜单中选择要导入的工程即可，选择好以后就直接next，在第二个窗口中也选择默认的第一个选项就可以。需要注意的是，在AS中，有两种工程，一个是Project，一个是Module，上面已经细说过了。
5. 导入jar包——选择File->Projcet Structure，在弹出的窗口中左侧找到Libraries并选中，然后点击“+”，并选择Java就能导入Jar包了。或者直接拷贝jar文件到项目的libs文件夹下，然后运行：Sync Project with Gradle Files。然后clean project重新编译。
6. 删除项目——AS对工程删除做了保护机制，默认你在项目右键发现没有删除选项。你会发现你的module上面会有一个小手机，这是保护机制。删除的第一步就是去掉保护机制，也就是让手机不见，具体做法就是鼠标放在工程上右键->open module setting，或者F4进入如图界面，选中你要删除的module，然后点击减号，这样就取消了保护机制，然后回到项目工程右键就可发现删除选项。注意：删除会将源文件删除。



1. 修改工程目录——在创建项目的时候，在Project Location中选好工程目录后，要自己输入一个文件夹的名字用来保存工程，然后就能使用自己的工程目录了。

入门总结

到此为止AS的基本情况相信你已经有个大致了解了。具体比Eclipse的优势体现在如下几点：

1. AS是Google专门为Android基于IntelliJ IDEA打造的利器。亲生的永远是最好的，只是现在还在成长中而已。
2. AS在速度上不管哪一个方面都比Eclipse快。
3. Darcula主题UI简直就是极客范，帅爆了。
4. 强大的智能提示补全功能在写代码时简直比Eclipse高效率N倍。
5. 智能保存，不需要Ctrl + S。效率会大大提升。
6. 整合Gradle构建工具，Gradle集合了Ant和Maven的优点，不管是配置、编译、打包都非常牛逼。
7. UI编辑器简直比Eclipse高效N倍，自带了多设备的实时预览，简直是神器。多语言适配点击地球直接输入，再也不用比较那个string没有翻译了。
8. 内置终端直接替代cmd命令行，一个IDE全部搞定。
9. 完善的插件系统，如Git、Markdown、Gradle等，直接搜索下载。
10. 版本控制系统，安装的时候就自带GitHub, Git, SVN等流行的版本控制系统，可以直接check out你的项目，边写代码边右键可以直接具备BCompare功能与其他版本进行对比修改。

总之就一句话，相信我，若果你和我一样是Eclipse用户切换过来，那么你绝对不会再切换回去，你会爱上AS的。

Android Studio目录结构

新建工程项目后AS的Product目录结构如下所示：

```
.idea：//AS生成的工程配置文件，类似Eclipse的project.properties。
app：//AS创建工程中的一个Module。
gradle：//构建工具系统的jar和wrapper等，jar告诉了AS如何与系统安装的gradle构建联系。
External Libraries：//不是一个文件夹，只是依赖lib文件，如SDK等。
```

- 1
- 2
- 3
- 4

- 1
- 2
- 3
- 4

新建工程项目后AS的Module目录结构如下所示：

```
build：//构建目录，相当于Eclipse中默认Java工程的bin目录，鼠标放在上面右键Show in Exploer即可打开文件夹，
    编译生成的apk也在这个目录的outs子目录，不过在AS的工程里是默认不显示out目录的，就算有编译结果也
    不显示，右键打开通过文件夹直接可以看。
libs：//依赖包，包含jar包和jni等包。
src：//源码，相当于eclipse的工程。
main：//主文件夹
    java：//Java代码，包含工程和新建是默认产生的Test工程源码。
    res：//资源文件，类似Eclipse。
        layout：//App布局及界面元素配置，雷同Eclipse。
        menu：//App菜单配置，雷同Eclipse。
        values：//雷同Eclipse。
            dimens.xml：//定义css的配置文件。
            strings.xml：//定义字符串的配置文件。
            styles.xml：//定义style的配置文件。
            ......：//arrays等其他文件。
        ......：//assets等目录
```

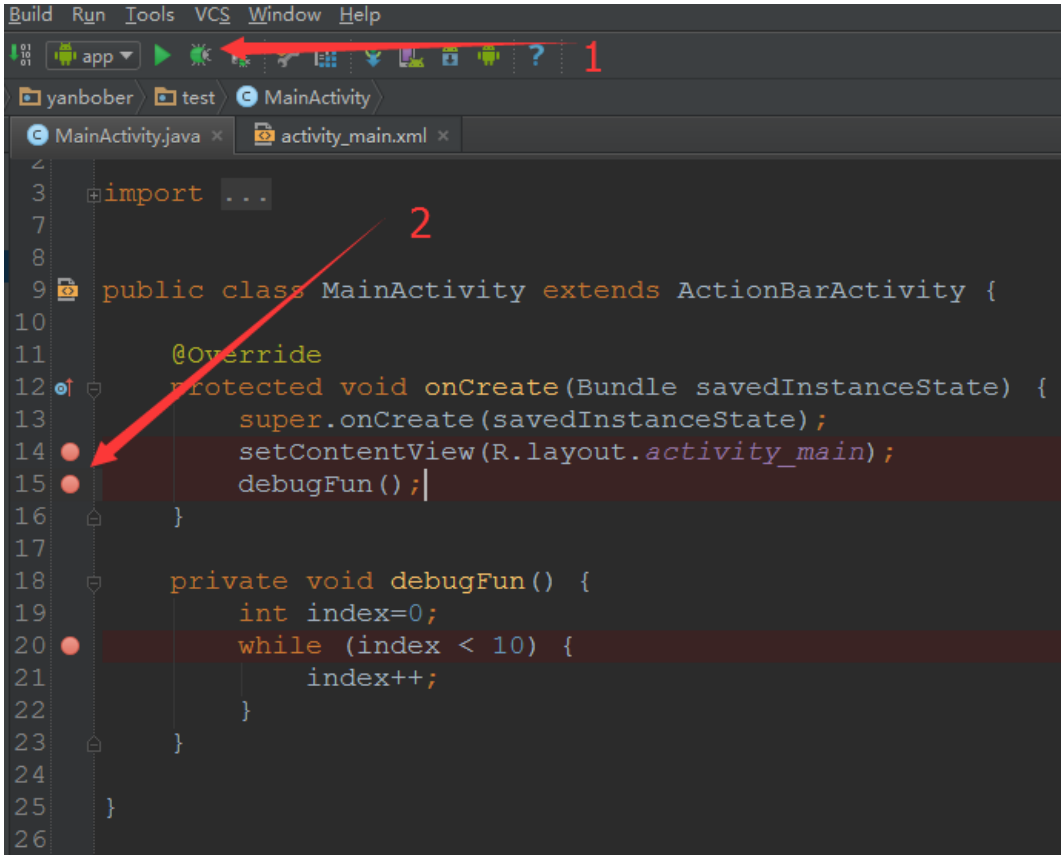
```
    AndroidManifest.xml：//App基本信息（Android管理文件）
    ic_launcher-web.png：//App图标
build.gradle：//Module的Gradle构建脚本
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
```

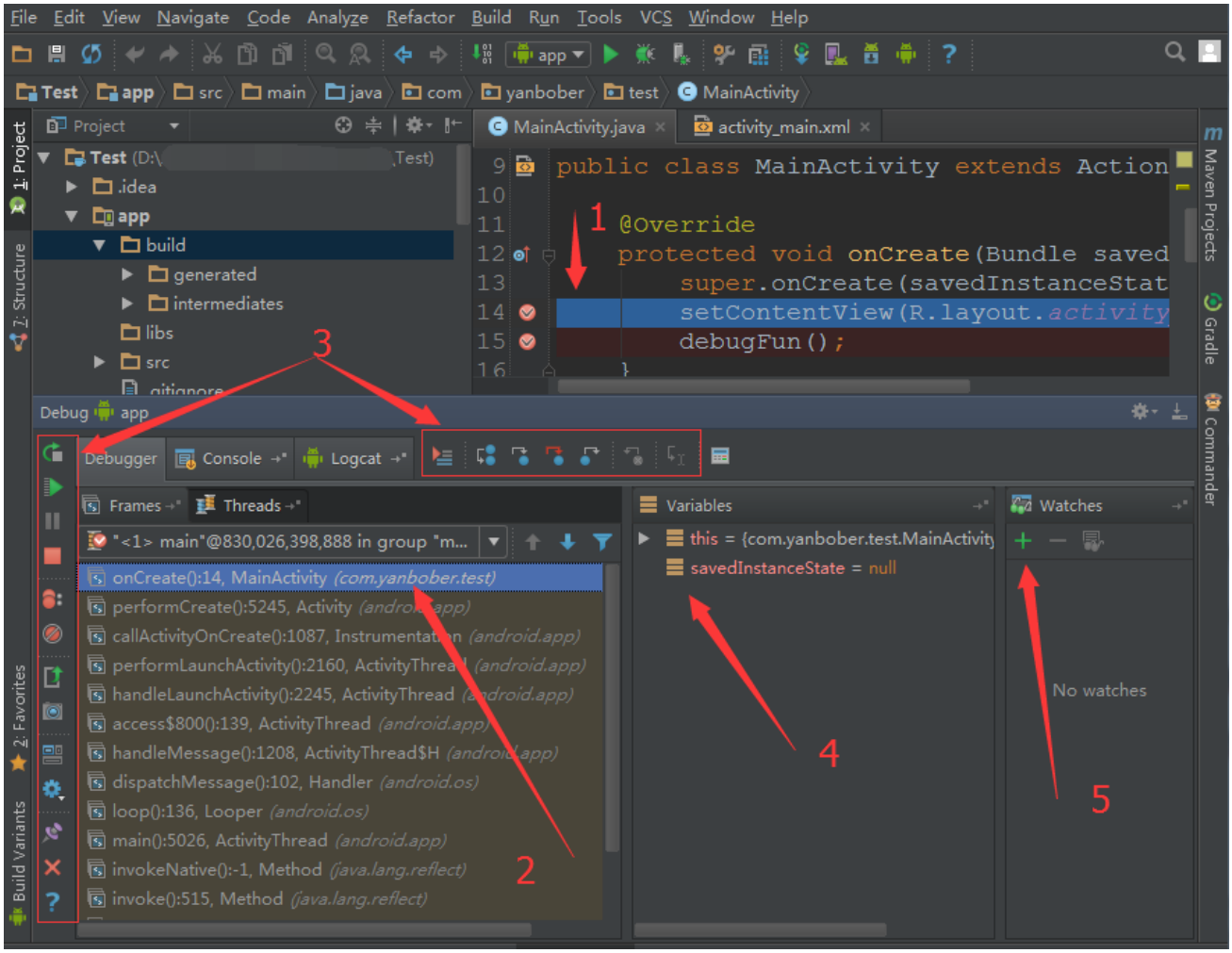
```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
```

Android Studio开发调试使用

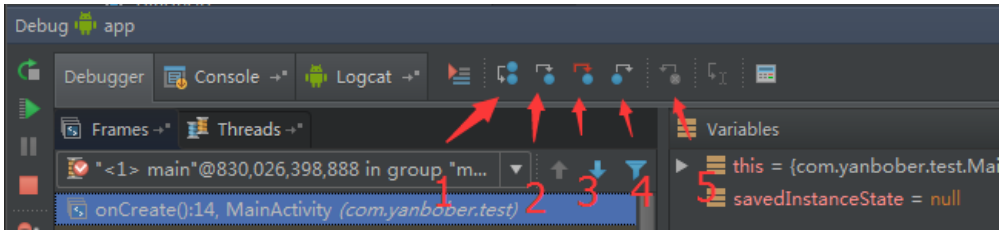
Android Studio调试其实也非常方便，一般问题直接通过AS的DDMS的Logcat就可以搞定。AS支持类似Eclipse的DDMS的所有功能。这里要说的是疑难问题的调试方式，即断点调试。
首先先编译好要调试的程序。



如上图2所示在行号处点击设置断点。然后如1所示点击开启调试会话。



如上图所示，IDE下方出现Debug视图，1指向的是现在调试程序停留的代码行，2区域是程序的方法调用栈区。在这个区域中显示了程序执行到断点处所调用过的所用方法，越下面的方法被调用的越早。由此顺序想必有些Android深入功底了解一点Android系统启动流程的就知道这几个方法咋回事，怎么到Activity的onCreate的。哈哈，说到系统了。不扯了。3是一些调试按钮，快捷键放在上面直接会显示。4和5是一些变量观察区。



上图中：

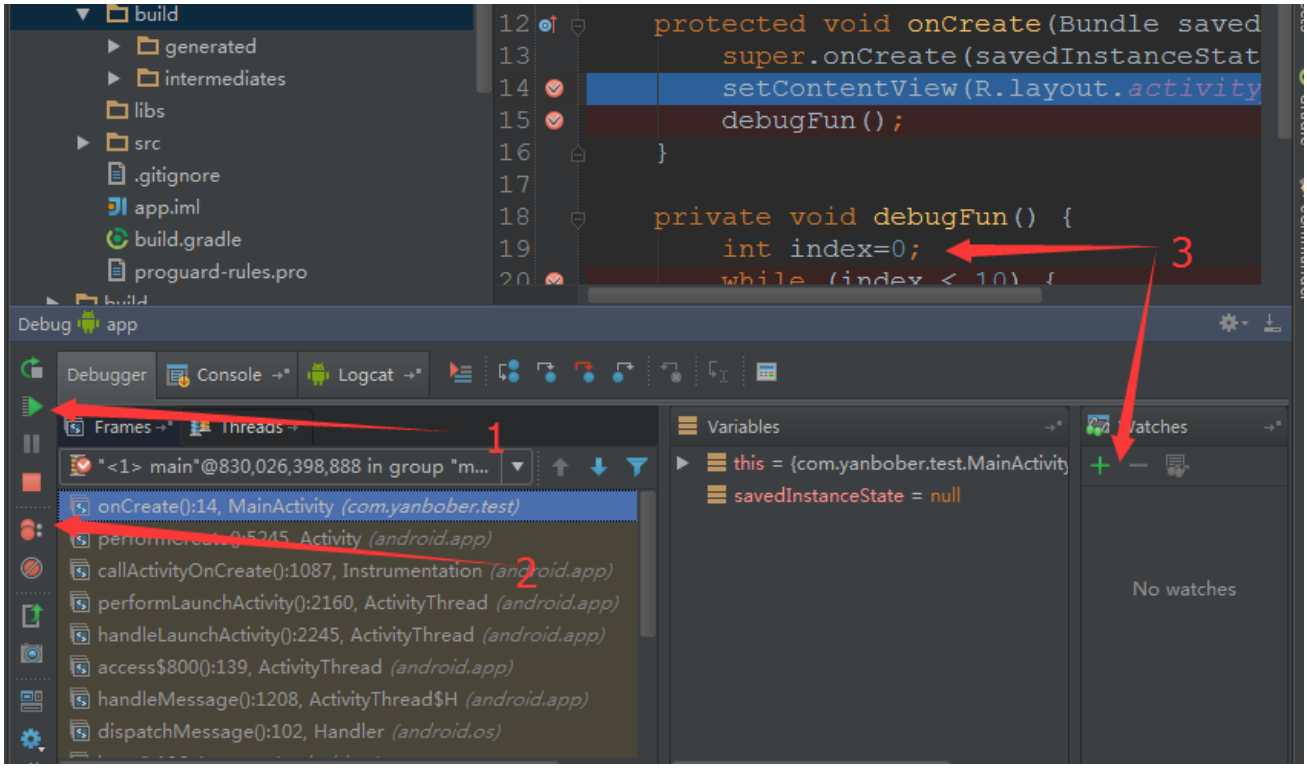
点击1指向的按钮，程序向下执行一行，如果当前行有方法调用，这个方法将被执行完毕返回，然后到下一行。

点击2指向的按钮，程序向下执行一行。如果该行有自定义方法，则运行进入自定义方法（不会进入官方类库的方法）。

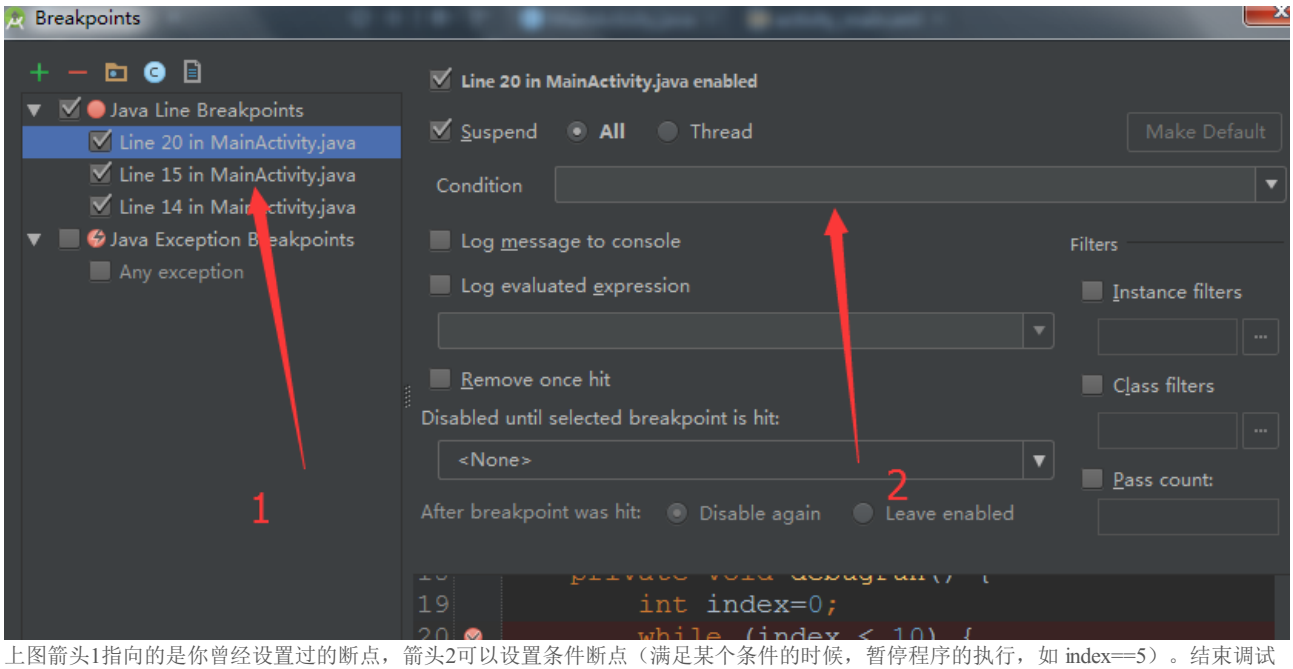
点击3钮在调试的时候能进入任何方法。

点击4的作用是如果在调试的时候你进入了一个方法(如debugFunc)，并觉得该方法没有问题，你就可以使用4跳出该方法，返回到该方法被调用处的下一行语句。值得注意的是，该方法已执行完毕。

点击5指向的按钮后，你将返回到当前方法的调用处重新执行，并且所有上下文变量的值也回到那个时候。只要调用链中还有上级方法，可以跳到其中的任何一个方法。



如上图设置多个断点，开启调试。想跨断点移动到下一个断点，点击如下图1箭头，程序将运行一个断点到下一个断点之间需要执行的代码。如果后面代码没有断点，再次点击该按钮将会执行完程序。点击箭头2指向的按钮，可以查看你曾经设置过的断点并可设置断点的一些属性，如下图所示。调试开始后，在Variables区域可以给指定的变量赋值（鼠标左键选择变量，右键弹出菜单选择setValue…）。这个功能可以更加快速的检测你的条件语句和循环语句。点击箭头3加号或者在调试的代码变量上右键添加watcher即可查看跟踪变量值。



上图箭头1指向的是你曾经设置过的断点，箭头2可以设置条件断点（满足某个条件的时候，暂停程序的执行，如 index==5）。结束调试后，应该在箭头1处把所设的断点删除(选择要删除的断点后，点击上方的红色减号）。

以上便是AS的一些调试使用技巧。

Android Studio构建系统基础

基础知识

项目创建成功后会自动下载Gradle，这个过程特别慢，建议翻墙。下载的Gradle在Windows平台会默认在 C:\Documents and Settings\<用户名>.gradle\wrapper\dists目录，这个目录下有个gradle-x.xx-all的文件夹,。也可以自己手动到Gradle官网下载对应的版本，然后将下载的.zip文件(也可以解压)复制到上述的gradle-x.xx-all 文件夹下。

每一个Module都需要有一个gradle配置文件，语法都是一样，唯一不同的是开头声明的是apply plugin。注意区分不同位置的build.gradle文件。

AS的工程根目录下的build.gradle文件：

```
buildscript {      //设置脚本的运行环境
    repositories { //支持java依赖库管理（maven/ivy等）,用于项目的依赖
        //mavenCentral()   //仅仅是不同的网络仓库而已
        jcenter()          //推荐使用这个仓库
    }
    //依赖包的定义。支持maven/ivy、远程、本地库、单文件，前面定义了repositories{}jcenter库，使用jcenter的依赖只需要按照
    //类似于com.android.tools.build:gradle:1.0.0-rc2，gradle就会自动的往远程库下载相应的依赖。
    dependencies {
        classpath 'com.android.tools.build:gradle:1.0.0-rc2'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}
//多项目的集中配置，多数构建工具，对于子项目的配置，都是基于继承的方式。Gradle除了提供继承方式设置子项目，还提供这种配置
allprojects {
    repositories {
        jcenter()
    }
}
```

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14

- 15
- 16
- 17
- 18
- 19
- 20

AS的工程根目录下的settings.gradle文件：

```
include ':app'      //module
include ':my_lib'   //module(build as lib)
```

- 1
- 2

- 1
- 2

AS的工程根目录下的Module的build.gradle文件（此处以一个简单的Lib module的gradle为例）：

```
//plugin在AS里取值一般为'com.android.library'或者'com.android.application'
apply plugin: 'com.android.library' //构建为lib

android {
    compileSdkVersion 17            //编译需要SDK版本
    buildToolsVersion "19.1.0"      //SDK Manager确定本地安装该版本才可以

    defaultConfig {
        minSdkVersion 8         //最小版本
        targetSdkVersion 17     //目标版本
    }

    buildTypes {                //编译项
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.txt'
        }
    }
}

dependencies {                  //依赖支持
    compile 'com.android.support:support-v4:18.+'
}
```

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23

- 1
- 2

Gradle打包APP签名

默认情况下，debug被配置成使用一个debug keystory。debug keystory使用了默认的密码和默认key及默认的key密码。debug构建类型会自动使用debug签名配置。在你的Module的build.gradle文件中添加：

```
android {
    ......
    signingConfigs {
      myConfig{
       storeFile file("yanbober.keystore")
          storePassword "gradle"
          keyAlias "gradle"
          keyPassword "gradle"
       }
    }

   buildTypes{
    release {
       runProguard true
       zipAlignEnabled true
       // 移除无用的resource文件
       shrinkResources true
       proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'

       signingConfig  signingConfigs.myConfig
    }
   }
}
```

```
18
19
20
21
22
23
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
```

虽然经常使用项目根目录的相对路径作为keystore的路径，但是也可以使用绝对路径，尽管这并不推荐（除了自动创建出来的debug keystore）。运行gradle clean gradle build即可生成签名混淆对齐的app。

Gradle构建Android应用多渠道包（批量打包）

Android应用的发布需要面对各种各样的市场，我们称之为渠道。通常作为开发者我们需要知道应用是从哪个渠道下载的。这种统计信息一般常用的是百度统计或者友盟统计。这里举例时使用友盟统计为例说明问题。原理是Gradle的Manifest Merger。

在AndroidManifest.xml里配置所谓的PlaceHolder。

```
<meta-data
    android:name="CHANNEL"
    android:value="${CHANNEL_VALUE}" />
```

```
1
2
3
```

```
1
2
3
```

在模块build.gradle文件的defaultConfig加上PlaceHolder，作用是声明CHANNEL_VALUE是可替换值的PlaceHolder，同时为其设置yanbober默认值。

```
android {
    ......

    defaultConfig {
        ......
        manifestPlaceholders = [ CHANNEL_VALUE:"yanbober" ]
    }
}
```

```
1
2
3
4
5
6
7
```

- 8

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8

在模块的build.gradle文件里添加ProductFlavors配置。ProductFlavors其实就是可定义的product特性，与Manifest Merger使用就可以在一次编译过程中产生多个具有自己特性配置的版本。下面这个配置的作用就是为每个渠道包产生不同的CHANNEL_VALUE的值。

```
android {
    ......

    defaultConfig {
        ......
        manifestPlaceholders = [ CHANNEL_VALUE:"yanbober" ]
    }
    productFlavors {
        yanbober{}
        wandoujia{}
        xiaomi{}
        baidu{}
    }
    productFlavors.all { flavor ->
        flavor.manifestPlaceholders = [ CHANNEL_VALUE:name ]
    }
}
```

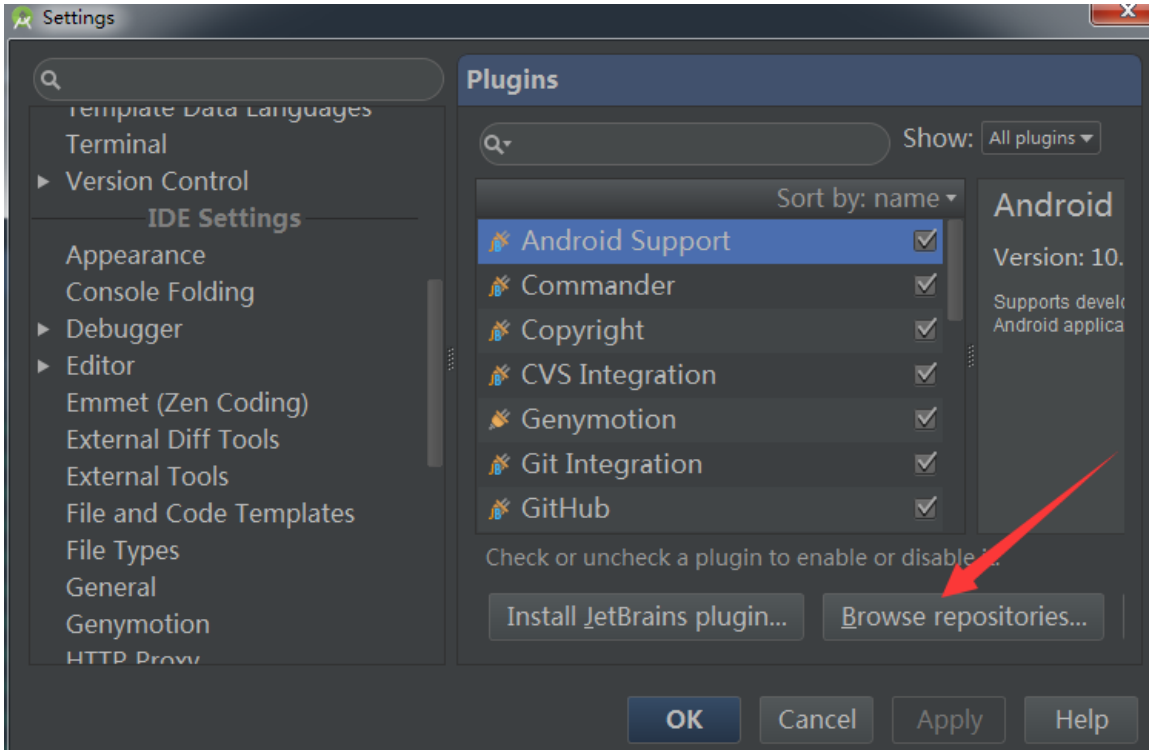
- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17

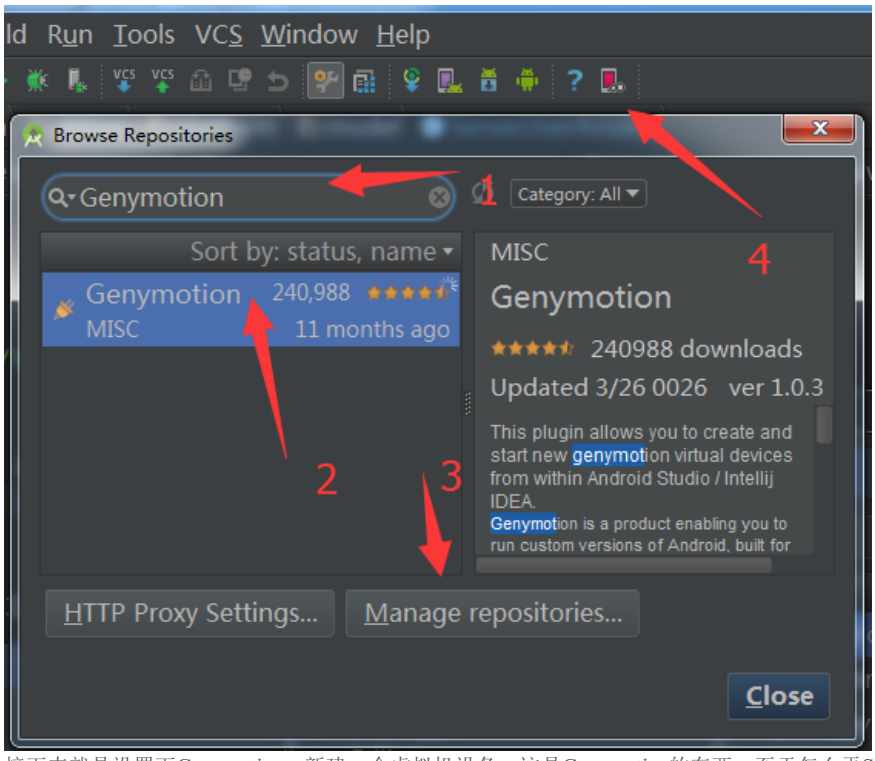
批量生成多渠道包：进入工程目录下运行gradlew assembleRelease。可以看到编译一共产生了4个apk，分别对应在productFlavors段定义的4个渠道。反编译打开AndroidManifest.xml就会发现CHANNEL这一段的配置已经被修改。
生成单个渠道包：打开AS的Gradle Tasks面板模块有很多任务，直接双击对应的耽搁渠道任务生成对应的apk。用命令行单独生成xiaomi渠道使用gradlew assemblexiaomiRelease就好了。
好了，Gradle的基本情况就说到这，具体可以阅读官网或者查阅其他资料，Gradle的使用需要经验的积累。

Android Studio插件安装及使用Genymotion模拟器

Android Studio自带的模拟器速度已经比Eclipse插件的快一点了，但是还不够暴力，不够爽。现在来说说最暴力的Genymotion模拟器如何结合AS使用。首先上Genymotion官网下载安装Genymotion，同时你需要在Genymotion官网官网上注册一个账号，这样你才能正常的使用Genymotion。
进入AS的Settings界面左侧找到Plugins，点击下图箭头Button（也就是在线安装）：



顺便说下上图界面也就是AS安装插件的通用方法，可以看见当前已经安装了的插件，选择在线安装或者从硬盘安装，即针对你已经下载好了的插件，可通过这项选择到你下好的插件，进行安装。
如下图所示在1区输入插件名字，2区选中，3区下载安装，然后返回后在AS工具栏上可以看见Genymotion小图标，也就是箭头4指的那个玩意，说明安装OK。



接下来就是设置下Genymotion，新建一个虚拟机设备，这是Genymotion的东西，至于怎么弄Step by Step就行，没啥难度。完事点击AS上模拟器图标就可以启动使用了，运行AS程序选择模拟器就可以在模拟器看见自己程序了，下图就是Genymotion启动起来的界面。

至此快速模拟器Genymotion已经搞定，提升你的速度。其他的插件安装也就触类旁通了。

其他

其他的也就是快捷键啥玩意的了。这东西就得自己积累慢慢整了，纯属积累熟练。诡异的问题就自行google和度娘了。其实到现在版本的AS还是有一些Bug的，但是满足基本需求了，遇见Bug查阅修改绕过或者使用大招—-重启AS一般就能解决，其他的诡异问题欢迎讨论共同成长。

PS：其他问题上AD就行了，这文章基本也就是AD的翻译版加上自己遇到的蛋疼问题的汇总了。

总之你会爱上他的。

Android Studio总结

到此你已经可以顺利使用Android Studio进行应用程序开发。其他的问题相信聪明的您使用Google可以搞定，祝你好运！总之AS的强大需要你自己去慢慢探索，你会发现不知不觉你会爱上他的。