Gradle For Android 故事集

一. 常见问题及解决

1.卡在refreshing gradle project

.....这个，点标题都有说明，自己看吧。

好吧，简单说一下，就是你的IDE没有能翻墙,所以自己去下覆盖下文件就好了，节约几个小时。

1. 找到gradle-wrapper.properties，位置在 项目根目录->gradle->wrapper->gradle-wrapper.properties
2. 复制distributionUrl的地址,从浏览器打开，注意把其中的"\"去掉才能下载，如下载gradle-2.14.1-all.zip
3. 把下载好的压缩包放到放到gradle系统目录,macos在~/.gradle/wrapper/dists/gradle-x.x.x-all
4. 把压缩包放到gradle-2.14.1-all这个文件夹目录下的随机生成的文件夹下即可
5. 重启Android Studio，试试看，不行自己搜去，谢谢

2. 引用本地aar

有时候我们需要引用别人的aar库,并不能像jar一样一行就解决

1.在项目build.gradle的根节点添加

```
repositories {
    flatDir {
        dirs 'libs'
    }
}
```

2.在dependencies内添加依赖

```
compile (name : 'zues-release', ext:'aar')
```

3. 引用冲突:不同的本地module引用相同的开源库

在老版本的Android Studio中,如果library module使用compile的方式引用开源库，最后会打包到aar， 后来更新2.x以上之后就没有这个问题了,使用compile或者provided都可以

4. Android Studio有提示的错误

有fix的，直接点fix就好了,不行再搜

二: Gradle的基石-Groovy

Gradle是基于groovy语言实现的插件,所以大概了解groovy可以比较有效的理解gradle脚本到底写的是些什么内容。groovy其实跟java基本相同,完全可以用Java的代码来写groovy。只是多了一些关键字和语法而已。语法跟js比较类似。

重点: 与java不同的一点是Groovy多了一个闭包的概念，跟oc里的代码块(block)是类似的东西。就是可以作为一个参数传递一个可执行的过程，gradle中大部分都是通过闭包来实现的。**闭包有一个隐式变量叫it,当然也可以通过->的方式进行变量的重命名**.

```
def hello = {
    println(it)
}
hello('hello wrold')
==> hello world

def sum = {
    a,b -> return a+b;
}
println(sum(3, 4))
==>7
```

现在我们再去看一下app下的build.gradle，看看是不是能看懂了

- 参考链接：http://www.jianshu.com/p/1e95d03060f7

三: 配置构建

配置构造是Google工程师扩展gradle,支持android更多的功能.具体有哪些我们通过官方文档来简单看一下。

重点的主要有

- buildTypes： 构建变体
- productFlavors：产品风格
- defaultConfig：默认产品配置，被所有产品版本共享，可以被覆盖,产品风格的结构与默认产品结构一样。

额外再说几个可能会有用的属性：

defaultPublishConfig：默认发布的构建变体, library项目默认是'release', 可以通过该属性来

修改

publishNonDefault : 没有默认发布的构建变体, 如果该属性设置为true,library会打包所有的构建产品

publishNonDefault与defaultPublishConfig理论上是互斥的关系,一种是只打包一个，另外一种是打包所有的构建产品

> *示范:*

库项目A的build.gradle中android内定义如下

```
android {
    defaultPublishConfig 'flavorRelease'
    compileSdkVersion 23
    buildToolsVersion '23.0.2'

    defaultConfig {
       minSdkVersion 9
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'),
'proguard-rules.pro'
        }

        hello {
            initWith(release)
        }
    }

    productFlavors {
        flavor{}
        flavor2{}
    }
}
```

App依赖库项目A不同的渠道包:

1.App区分flavor3, flavor4，分别依赖库项目的flavor,flavor2，那dependencies的写法就是如下这样:

```
    dependencies {
        flavor3Compile project(path:':A', configuration:'flavorRelease')
        flavor4Compile project(path:':A', configuration:'flavor2Release')
    }
```

Tips: *注意需要区分Release还是Debug,一般依赖的aar都使用release吧。并且库项目* publishNonDefault为true_

2.如果App也细致的区分productFlavors和buildTypes所依赖的构建产品，那么，我们还需要再做一步，在dependencies同级再重写一个闭包configurations，如下:

```
    configurations {
        flavor3ReleaseCompile
        flavor4ReleaseCompile
    }
```

这样定义之后,dependencies变成如下:

```
    dependencies {
        flavor3ReleaseCompile project(path:':A',
configuration:'flavorRelease')
        flavor4ReleaseCompile project(path:':A',
configuration:'flavor2Release')
    }
```

flavorDimensions:产品风格还可以再细分多个维度,这个有顺序,定义在前面的，源集名就在前面，跟productFlavors在buildTypes一样。

sourceSets: 源集，定义buildTypes与productFlavors就生成了相关源集的配置，可以通过新建如下的文件夹来进行构建产品的自定义。

src/main/ 此源集包括所有构建变体共用的代码和资源。

src/<buildType>/ 创建此源集可加入特定构建类型专用的代码和资源。

src/<productFlavor>/ 创建此源集可加入特定产品风味专用的代码和资源。

src/<productFlavorBuildType>/ 创建此源集可加入特定构建变体专用的代码和资源。

他们会进行一层层覆盖,优先级 : 构建变体 > 构建类型 > 产品风味 > 主源集 > 库依赖项

consumerProguardFiles: 库项目保留的混淆文件,省去了去主项目配置混淆的麻烦。

看的过程中我接触到的新知识:

Data Binding入门篇：http://blog.zhaiyifan.cn/2016/06/16/android-new-project-from-0-p7/

参考链接 :

- android 配置构建 官方说明 : https://developer.android.com/studio/build/index.html
- android 构建变体 官方说明: https://developer.android.com/studio/build/build-variants.html
- android 库说明: https://developer.android.com/studio/projects/android-library.html?hl=zh-cn
- android gradle dsl: http://google.github.io/android-gradle-dsl/current/index.html
- android gradle dsl in github :https://github.com/google/android-gradle-dsl
- 多构建产品依赖: http://wiki.jikexueyuan.com/project/android-gradle-guide/libraries-and-multi-project-setup.html

---

四: 10分钟实现Maven私有库

就像松科构思的是，想把公司的android项目，能共用的就共用，但是，现在的情况是，触手录跟触手tv是不同的项目，对于单独一个项目来说，svn可以管理你所有的代码，但是对于另一个项目来说，用另外一个项目的公用部分就有点麻烦了，一个是沟通，一个是实现起来，难道再有一个新项目，又要强制去你那个项目去拷贝这个库吗？这并不现实。

其实目的就是公有库的使用，类似于jcenter的功能，只是不想给外网能够使用，那我们自己建立一个maven库就可以了，那以后通过compile服务器的库，就可以了。也便于其他各个项目去依赖。形成公司自有的代码库。

1. 下载nexus开源项目
2. 运行bin/nexus start
3. 查看自己ip，浏览器打开ip:8081/nexus,端口可以修改conf/nexus.properties，默认admin帐号是admin/admin123
4. 可以使用默认的库，也可以自己再添加,随意，帐号也可以新添加
5. 配置Android Studio uploadArchives[1],注意不要使用publishNonDefault为true,否则会发布所有的构建产品的aar包,如果需要可以通过写groovy来实现遍历所有变体，结合defaultPublishConfig和修改上传的artifactId，再上传
6. 浏览器上可以看到上传的下面有aar包
7. 在需要用的项目中依赖[2]

[1]

```
uploadArchives {
        repositories {
            mavenDeployer {
                snapshotRepository (url : ${snapshotUrl}){
                    authentication(userName: ${name}, password: ${pwd})
                }
                repository (url : ${url}){
                    authentication(userName: ${name}, password: ${pwd})
                }
                pom.project {
                    version : ${version}
                    groupId : ${groupId}
                    artifactId : ${artifactId}
                    packing : aar/jar, default is jar
                    description : 描述该版本
                }
            }
        }
    }
```

[2]

```
    // 看需要添加私有maven添加到repositories节点下
    repositories {
        jcenter()
        maven { url MAVEN_URL }
    }

    // 在项目中跟使用jcenter的库一样使用自己的库
    compile ${groupId}:${artifactId}:${version}
```

参考链接:

- http://zmywly8866.github.io/2016/01/05/android-private-maven-repository.html

五: 集成环境Jenkins

在以前的公司，有个同事把很繁杂的工作交给了他以前使用过的一个集成环境，Jenkins，可以执行很多自动化的工作，节省很多时间，这个在后期也可以考虑集成。以后只需要打开网站，点几下，就可以把以前需要自己动手的很多事情完成。而且可以让大多数人都能用，降低学习门槛。

六: 补充学习资料

在查找资料的过程中，发现极客学院免费的百科，可以比较全面的补充知识面。

http://wiki.jikexueyuan.com/list/android/