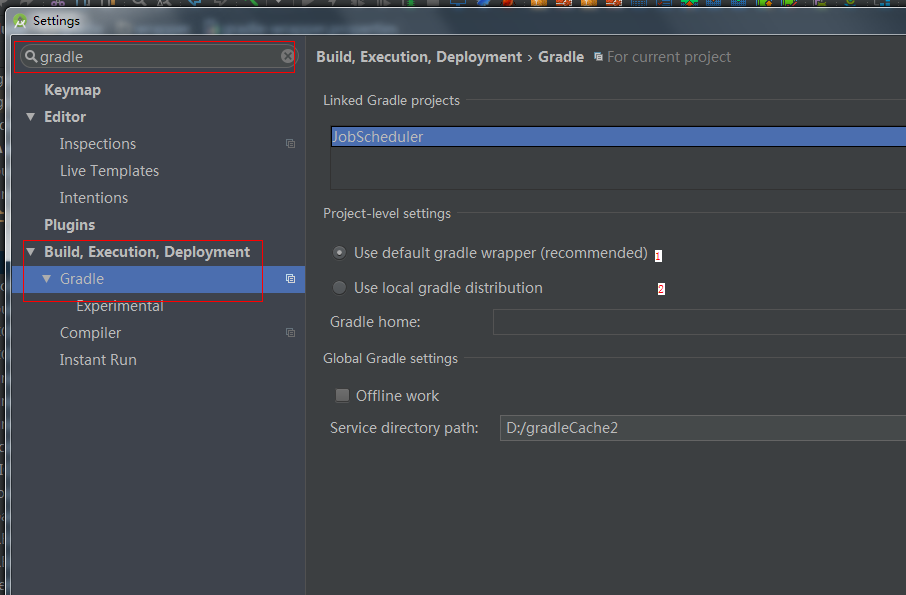
Gradle Guide

* **初步介绍**

1.安装Gradle(以windows为例)

方法一：下载Gradle，解压，配置bin目录到环境变量到PATH

方法二：使用Android Studio，可以使用Gradle Wrapper。在gradle/wrapper/gradle-wrapper.properties路径下，更改distributionUrl的值，获取不同版本的Gradle Wrapper



1……………使用gradle wrapper

2……………使用本地的gradle

2.新的Gradle构建

初始化项目

>mkdir gradle-demo

>cd gradle-demo

>gradle init

目录结构：

├── build.gradle…………………………………1

├── gradle

│   └── wrapper

│   ├── gradle-wrapper.jar…………………2

│   └── gradle-wrapper.properties…………3

├── gradlew…………………………………………………4

├── gradlew.bat………………………………………………5

└── settings.gradle……………………………………………6

1.project的全局配置脚本，用来配置任务

2.Gradle Wrapper的执行jar包

3.Gradle Wrapper属性配置文件

4.Unix-based 操作系统的Gradle Wrapper脚本

5.Windows操作系统的Gradle Wrapper脚本

6.Settings配置脚本，用来配置Project中参与编译的模块

3.Task

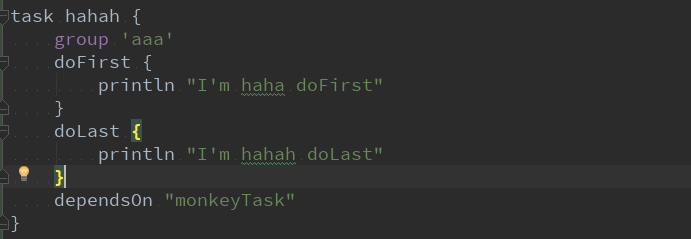
文档：<https://docs.gradle.org/current/dsl/org.gradle.api.Task.html>

·添加一个plugin到构建文件中将会自动构建一系列的任务。Gradle是属于任务驱动型的构建工具，构建过程是基于Task。

查看所有的任务

>gradlew tasks --all（gradle tasks）

·除了使用内置任务，还可以自建任务。



>gradlew hahah(gradle hahah)

>gradlew properties（gradle properties）

·Task基本属性介绍

dependsOn:设置依赖的任务

group:设置任务的分组

name:任务的名字

project:任务所属的project

doLast:将action添加到任务的action列表的结尾

doFirst:将action添加到任务的action列表的开始

* **gradle与插件**

Gradle是一个通过插件形式自定义构建逻辑的构建工具。

1.Android构建插件

* app模块的需要的插件, apply plugin: ‘com.android.application’
* library模块需要的插件，apply plugin: ‘com.android.labrary’

2.Java构建插件

* Java插件：apply plugin: ‘java’
* Java Library插件：apply plugin: ‘java-library’

3代码质量插件

* Checkstyle插件：apply plugin: ‘checkstyle’

<https://docs.gradle.org/current/userguide/checkstyle_plugin.html>

* FindBugs插件：apply plugin: ‘findbugs’

<https://docs.gradle.org/current/userguide/findbugs_plugin.html>

* JoCoCo插件(代码覆盖率)：apply plugin: ‘jacoco’

<https://docs.gradle.org/current/userguide/jacoco_plugin.html>

* JDepend插件（Java软件包结构质量进行分析）：apply plugin: ’jdepend’

<https://docs.gradle.org/current/userguide/jdepend_plugin.html>

* PMD插件（通过静态分析获知代码错误）：apply plugin：’pmd’

<https://docs.gradle.org/current/userguide/pmd_plugin.html>

4自定义Gradle插件

<https://guides.gradle.org/designing-gradle-plugins/>

<http://kvh.io/cn/embrace-android-studio-gradle-plugin.html>

* **Gradle构建--DSL（Domain Specific Lauguage:领域定义语言）**

**官方文档：**[**https://docs.gradle.org/current/dsl/**](https://docs.gradle.org/current/dsl/)

1.脚本的代理对象

| Type of script | Delegates to instance of |
| --- | --- |
| Build script | [Project](https://docs.gradle.org/current/dsl/org.gradle.api.Project.html) |
| Init script | [Gradle](https://docs.gradle.org/current/dsl/org.gradle.api.invocation.Gradle.html) |
| Settings script | [Settings](https://docs.gradle.org/current/dsl/org.gradle.api.initialization.Settings.html) |

2.脚本块

| Block | Description |
| --- | --- |
| [allprojects { }](https://docs.gradle.org/current/dsl/org.gradle.api.Project.html#org.gradle.api.Project:allprojects(groovy.lang.Closure)) | 配置项目和子项目 |
| [artifacts { }](https://docs.gradle.org/current/dsl/org.gradle.api.Project.html#org.gradle.api.Project:artifacts(groovy.lang.Closure)) | 配置项目的发行物 |
| [buildscript { }](https://docs.gradle.org/current/dsl/org.gradle.api.Project.html#org.gradle.api.Project:buildscript(groovy.lang.Closure)) | 配置项目构建脚步的classpath |
| [configurations { }](https://docs.gradle.org/current/dsl/org.gradle.api.Project.html#org.gradle.api.Project:configurations(groovy.lang.Closure)) | 配置项目的依赖配置 |
| [dependencies { }](https://docs.gradle.org/current/dsl/org.gradle.api.Project.html#org.gradle.api.Project:dependencies(groovy.lang.Closure)) | 配置依赖 |
| [repositories { }](https://docs.gradle.org/current/dsl/org.gradle.api.Project.html#org.gradle.api.Project:repositories(groovy.lang.Closure)) | 配置以来仓库 |
| [sourceSets { }](https://docs.gradle.org/current/dsl/org.gradle.api.Project.html#org.gradle.api.Project:sourceSets(groovy.lang.Closure)) | 配置项目的源集 |
| [subprojects { }](https://docs.gradle.org/current/dsl/org.gradle.api.Project.html#org.gradle.api.Project:subprojects(groovy.lang.Closure)) | 配置当前项目的子项目 |
| [publishing { }](https://docs.gradle.org/current/dsl/org.gradle.api.Project.html#org.gradle.api.Project:publishing(groovy.lang.Closure)) | 配置发行插件的 [PublishingExtension](https://docs.gradle.org/current/dsl/org.gradle.api.publish.PublishingExtension.html) |

* **构建Android APP**

1.根层级的build.gradle文件

// Top-level build file where you can add configuration options common to all sub-projects/modules.

buildscript {

repositories {...............................1

google()

jcenter()

}

dependencies {...............................2

classpath 'com.android.tools.build:gradle:3.0.1'

// NOTE: Do not place your application dependencies here; they belong

// in the individual module build.gradle files

}

}

allprojects {............................................3

repositories {

google()

jcenter()

}

}

task clean(type: Delete) {.......................4

delete rootProject.buildDir

}

|  |
| --- |
|  |

1.classpath的依赖仓

2.gradle android 插件

3.根层级配置，模块配置

4.clean 任务

2.app模块的build.gradle

apply plugin: 'com.android.application'

android {

compileSdkVersion 26

defaultConfig {

applicationId "org.gradle.helloworldgradle"

minSdkVersion 19

targetSdkVersion 26

versionCode 1

versionName "1.0"

testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"

}

buildTypes {

release {

minifyEnabled false

proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'

}

}

}

3.library模块的build.gradle

apply plugin: 'com.android.library'

其他没有和app模块的build.gradle没有区别

4.运行Gradle任务

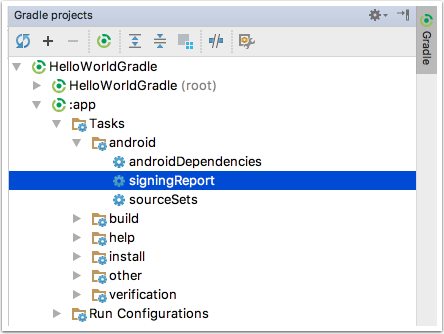
4.1命令行

>gradlew build (gradle build)

>gradlew assembleDebug (gradle assembleDebug)

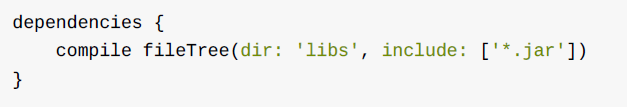
>gradlew assembleRelease (gradle assembleRelease)

4.2使用Gradle任务窗口



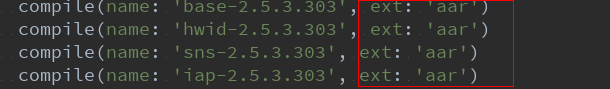
5.依赖

5.1本地jar包依赖



5.2本地aar包依赖





5.3远程包依赖



5.4依赖其他模块



6.变体(Variant)

6.1变体的作用是，为同一个应用创建不同的版本

Variant(变体) = buildTypes(构建类型) \* ProductFlovars(产品风味)

eg:

buildTypes: debug/release

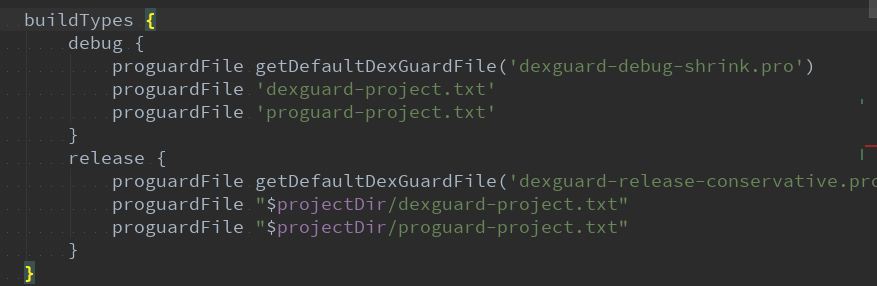
ProductFrovars:\_test/\_develop/\_online

Variant:\_testDebug/\_testRelease

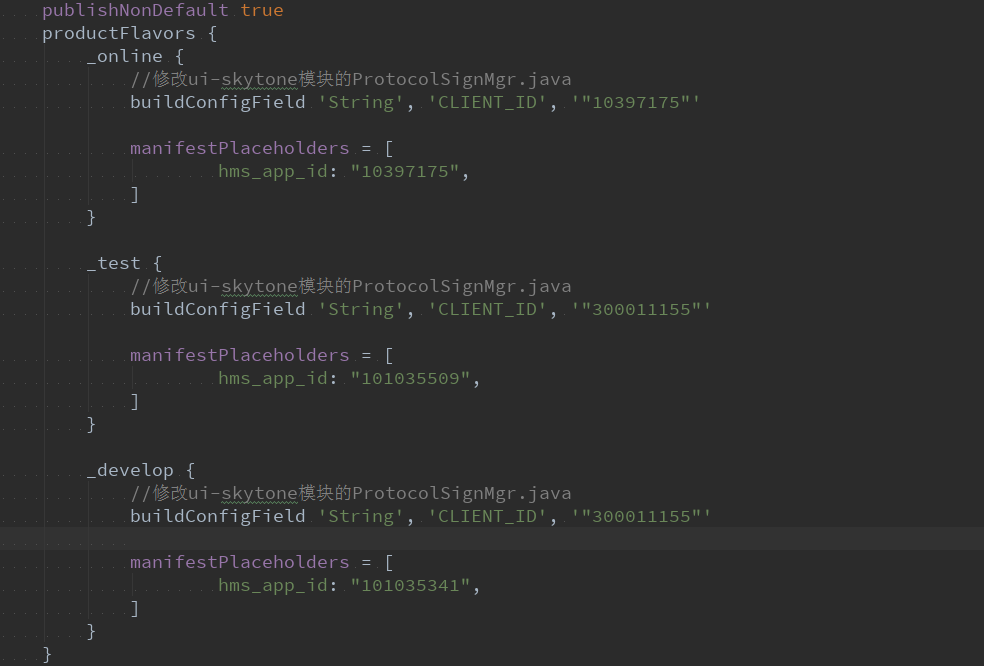
\_developDebug/\_developRelease

\_onlineDebug/\_onlineRelease

6.2构建类型



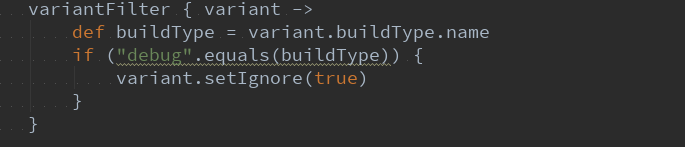
6.3产品风味



1）每个变体都有一个自动生成文件BuildConfig.java. buildConfigField会在BuildConfig文件中自动生成一个Field，我们可以可以借助这个机制，实现不同环境的风味定制。

2）manifestPlaceholders可以根据不同产品风味来定制AndroidManifest.xml文件

6.4 过滤变体



* **迁移到Android Gradle3.+**

官方引导说明指导：<https://developer.android.com/studio/build/gradle-plugin-3-0-0-migration>