# Maven公有库发布流程

## 混淆代码

纯Java或包含核心功能及Java代码需要使用深度混淆功能，防止外部组织/公司轻易破解，建议核心层代码能使用C、C++代码替换的，使用C等语言打包成二进制SO库，或者使用Jni反射机制。

## 项目仓库配置

1、在Github或Gitlab上面创建Public公有库，比如JimiTest：<https://github.com/JimiPlatform/JimiTest>

2、仓库必须包含发布的SDK项目代码、[LICENSE](https://github.com/JimiPlatform/JimiOrderCoreKit/blob/master/LICENSE)、[README.md](https://github.com/JimiPlatform/JimiOrderCoreKit/blob/master/README.md)；

3、若SDK的源码不能公开，第2点项目代码可以以SDK名称创建一个简单初始化的Android项目；

## Module发布配置

**1、在SDK Module中的build.gradle配置JFrog Bintray设置**：

apply plugin: 'com.android.library'

apply plugin: 'com.github.dcendents.android-maven'

apply plugin: 'com.jfrog.bintray'

def Jimi\_SDK\_Version\_Value = "1.0.0" //SDK版本号

android {

……

defaultConfig {

//……

versionName Jimi\_SDK\_Version\_Value

//……

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Maven仓库发布

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

buildscript {

repositories {

google()

jcenter()

}

dependencies {

classpath 'com.android.tools.build:gradle:3.4.2' //根据自己的AS配置Gradle编译工具版本号

classpath 'com.jfrog.bintray.gradle:gradle-bintray-plugin:1.7.3'

classpath 'org.jfrog.buildinfo:build-info-extractor-gradle:4.0.0'

classpath 'com.github.dcendents:android-maven-gradle-plugin:2.1'

}

}

def siteUrl ='<https://github.com/JimiPlatform/JimiTest>' //根据[项目仓库配置](#_项目仓库配置)中的公有库进行设置

def gitUrl ='<https://github.com/JimiPlatform/JimiTest>.git' //根据[项目仓库配置](#_项目仓库配置)中的公有库进行设置

group = "com.jimi" //不要修改

def libName = "JimiTest" //这是SDK的名称

version = Jimi\_SDK\_Version\_Value

task sourcesJar(type: Jar) {

from android.sourceSets.main.java.srcDirs

classifier = 'sources'

}

task javadoc(type: Javadoc) {

source = android.sourceSets.main.java.srcDirs

classpath += project.files(android.getBootClasspath().join(File.*pathSeparator*))

options.encoding "UTF-8"

options.charSet 'UTF-8'

options.author true

options.version true

failOnError false

}

task javadocJar(type: Jar, dependsOn: javadoc) {

classifier ='javadoc'

from javadoc.destinationDir

}

artifacts {

archives javadocJar

archives sourcesJar

}

install {

repositories.mavenInstaller {

pom {

project {

packaging 'aar'

name 'Jimi Test for Android Maven' //SDK描述

url siteUrl

licenses {

license {

name 'The Apache Software License, Version 2.0'

url 'http://www.apache.org/licenses/LICENSE-2.0.txt'

}

}

developers {

developer {

id 'eafy'

name 'jason'

email 'lizhijian\_21@163.com'

}

}

scm {

connection gitUrl

developerConnection gitUrl

url siteUrl

}

}

}

}

}

Properties properties = new Properties()

properties.load(project.rootProject.file('local.properties').newDataInputStream())

bintray {

user = properties.getProperty("bintray.user") //读取 local.properties 文件里面的bintray.user

key = properties.getProperty("bintray.apikey") //读取 local.properties 文件里面的bintray.apikey

configurations = ['archives']

pkg {

repo = "maven"

name = libName

desc = ' Jimi Test for Android Maven'' //项目描述

websiteUrl = siteUrl

vcsUrl = gitUrl

licenses = ["Apache-2.0"]

publish = true

}

}

**2、配置上传开发者信息**

在工程的local.properties中配置/添加如下参数：

bintray.user=eafy

bintray.apikey=\*\*\*\*\*\*\* #JFrog Bintray开发者信息，找对应的管理员获取

## 发布上架

1、工程编译通过之后，修改**build.gradle**文件中的Jimi\_SDK\_Version\_Value版本号，即发布的版本号；

2、展开右侧Gradle工具栏，找到对应Module下的publishing，展开并双击bintrayUpload，进行上传操作；

3、等待上传成功之后，打开<https://bintray.com/>并登陆；

4、打开对应的Repository仓库，比如maven；

5、打开对应的SDK项目，在左下角点击Add to Jcenter，调整连接之后直接点击send即可。

## SDK使用/引用

在dependencies中引用SDK:

dependencies {

//……

implementation ‘com.jimi.JimiTest:1.0.0’ //即com.jimi:SDK名称:SDK版本号，对应配置中的group:libName: Jimi\_SDK\_Version\_Value

}