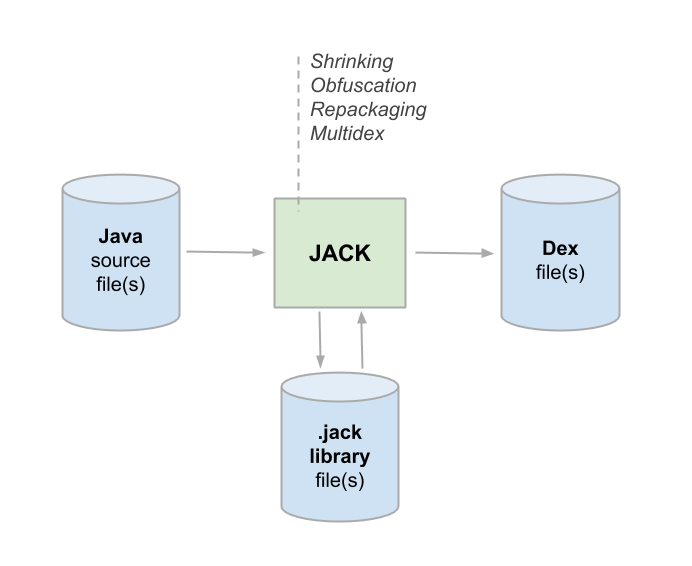
Overview

Jack (Java Android Compiler Kit)是新的Android 编译工具，从Android 6.0 开始加入,替换原有的编译工具，例如javac, ProGuard, jarjar和 dx。它主要负责将java代码编译成dex包，并支持代码压缩，混淆等

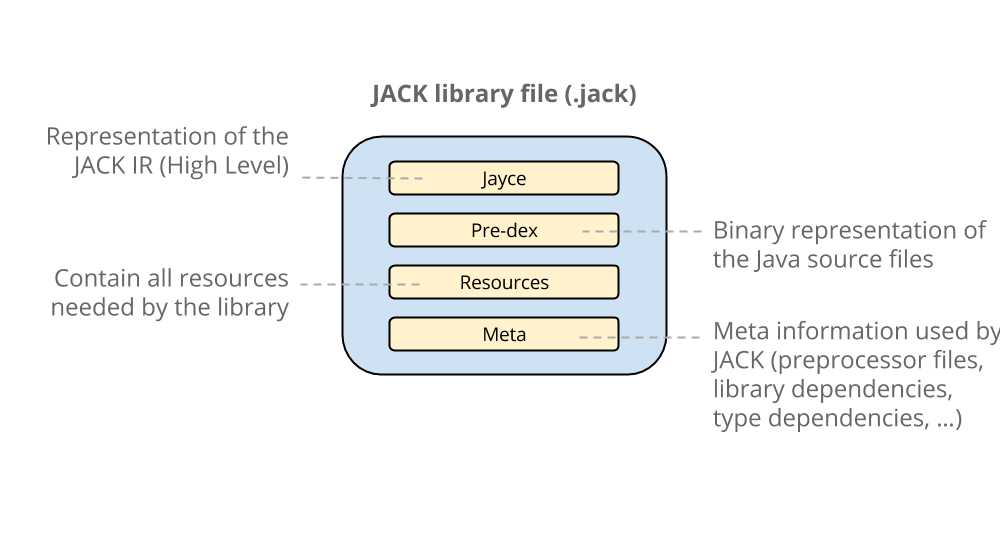
Jack工具的主要优势

完全开放源码  
源码均在AOSP中，合作伙伴可贡献源码

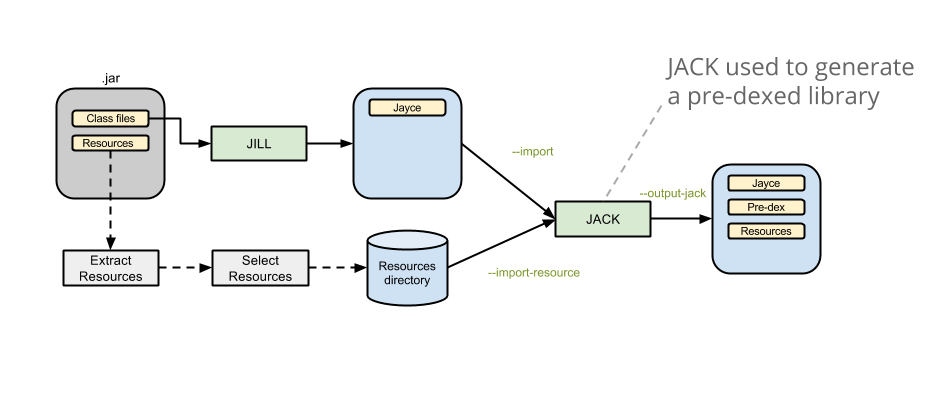
加快编译源码  
Jack 提供特殊的配置，减少编译时间：pre-dexing, 增量编译和Jack编译服务器.

支持代码压缩，混淆，重打包和multidex  
不在使用额外单独的包，例如ProGuard。  


The .jack library format

Jack 有单独的文件格式.jack, 它包含有预编译的dex代码，允许快速编译（pre-dex）.  


Jack Intermediate Library Linker (Jill)

Jill 工具能够将jar包转换成jack格式，转换过程如下  


使用Jack 编译Android

当你编译Android时，你不需要修改任何内容。 Jack是Andriod M的默认编译工具。只需使用标准的makefile命令执行即可。  
当第一次执行jack时，它会在你的机器上启动一个jack编译服务：  
这个服务大大提高了编译速度，因为它避免了启动一个新的JVM，加载Jack代码，初始化Jack和每次编译都要预制JIT，即使在很小的编译过程中，它也很好的节省了时间（例如，在incremental模式）  
这个服务也是一个控制Jack并行编译的短期解决方案。它可以限制编译的并行数据量，这样也可以避免内存硬盘溢出问题  
当一段时间没有编译活动，Jack 服务会自动关闭，默认它在本地使用了两个TCP端口，且不对外公开。它也提供了很多参数来配置Jack服务，例如并行数量，timeout时间，端口等等。只修改$HOME/.jack 文件即可。

$HOME/.jack 文件

$HOME/.jack 文件包含了Jack服务的设置变量

以下是现在可用的变量说明：

SERVER=true 启用Jack serverSERVER\_PORT\_SERVICE=8072 编译服务TCP端口.SERVER\_PORT\_ADMIN=8073 管理服务TCP端口SERVER\_COUNT=1 目前没用SERVER\_NB\_COMPILE=4 最大编译并行数SERVER\_TIMEOUT=60 空闲时间分钟数据，无编译时间超出该时间，将自动关闭服务SERVER\_LOG=${SERVER\_LOG:=$SERVER\_DIR/jack-$SERVER\_PORT\_SERVICE.log} 服务器日志存放目录，可被环境变量替换JACK\_VM\_COMMAND=${JACK\_VM\_COMMAND:=java} 默认启动JVM的命令，可被环境变量替换

Jack 问题

你的电脑没有响应或者编译过程出现内存溢出  
你可以减少SERVER\_NB\_COMPILE数量，来减少编译占用内存。

Cannot launch background server  
很有可能时TCP端口被占用，试着修改SERVER\_PORT\_SERVICE和SERVER\_PORT\_ADMIN参数

Jack 限制

Jack server 默认是单用户的，所以只能同时被一个用户使用。如果需要多个用户同时使用，请每个人都使用不同的端口。也可以通过设置SERVER=false 禁用Jack Server

CTS 编译会变慢.

二进制码有关的工具将不支持, 例如 JaCoCo.

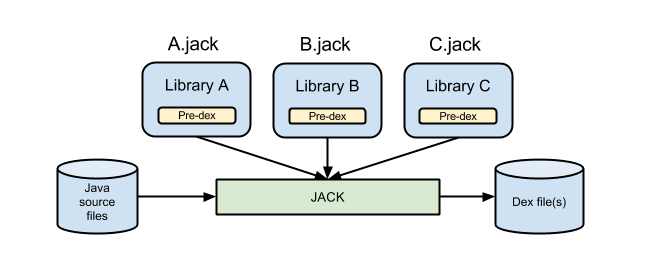
使用Jack特性

Jack支持JDK1.7，并且整合了一下特性

Predexing

当生成Jack包时，dex包将会被生成，并存储到Jack包中，这个过程就是pre-dex，在编译过程中，Jack会为每一个包使用pre-dex.

所有的包都将pre-dexed



限制：

当前版本，如果设置了代码压缩/混淆/重新打包，Jack将不会重用pre-dex 包

Incremental

Incremental compilation means that only components that were touched since the last compilation, and their dependencies, are recompiled. Incremental compilation can be significantly faster than a full compilation when changes are limited to only a limited set of components.

限制：

Incremental compilation is deactivated when shrinking, obfuscation, repackaging or multi-dex legacy is enabled.

Enabling incremental builds

Currently incremental compilation is not enabled by default. To enable incremental builds, add the following line to the Android.mk file of the project that you want to build incrementally:

LOCAL\_JACK\_ENABLED := incremental  
Note: The first time that you build your project with Jack if some dependencies are not built, use mma to build them, and after that you can use the standard build command.

压缩和混淆

JACK has shrinking and obfuscation support and uses proguard configuration files to enable shrinking and obfuscation features. Here are the supported and ignored options:

Supported common options

Common options include the following:

@  
-include  
-basedirectory  
-injars  
-outjars // only 1 output jar supported  
-libraryjars  
-keep  
-keepclassmembers  
-keepclasseswithmembers  
-keepnames  
-keepclassmembernames  
-keepclasseswithmembernames  
-printseeds  
Supported shrinking options

Shrinking options include the following:

-dontshrink

Supported obfuscation options

Obfuscation options include the following:

-dontobfuscate  
-printmapping  
-applymapping  
-obfuscationdictionary  
-classobfuscationdictionary  
-packageobfuscationdictionary  
-useuniqueclassmembernames  
-dontusemixedcaseclassnames  
-keeppackagenames  
-flattenpackagehierarchy  
-repackageclasses  
-keepattributes  
-adaptclassstrings

Ignored options

Ignored options include the following:

-dontoptimize // Jack does not optimize  
-dontpreverify // Jack does not preverify  
-skipnonpubliclibraryclasses  
-dontskipnonpubliclibraryclasses  
-dontskipnonpubliclibraryclassmembers  
-keepdirectories  
-target  
-forceprocessing  
-printusage  
-whyareyoukeeping  
-optimizations  
-optimizationpasses  
-assumenosideeffects  
-allowaccessmodification  
-mergeinterfacesaggressively  
-overloadaggressively  
-microedition  
-verbose  
-dontnote  
-dontwarn  
-ignorewarnings  
-printconfiguration  
-dump  
Note: Other options will generate an error.

Repackaging

JACK 使用 jarjar 来实现重新打包

Note: JACK is compatible with "rule" rule types, but is not compatible with "zap" or "keep" rule types. If you need "zap" or "keep" rule types please file a feature request with a description of how you use the feature in your app.

Multidex support

Since dex files are limited to 65K methods, apps with over 65K methods must be split into multiple dex files. (See ‘Building Apps with Over 65K Methods’ for more information about multidex.)

Jack offers native and legacy multidex support.