免費電子書

學習

gradle

Free unaffiliated eBook created from
Stack Overflow contributors.

#gradle

1: gradle

Gradle。 JavaAndroid 。

Gradle

- GroovyKotlin。
-
- 。
- MavenIvy。 `repositories`npm。
- 。
- Maven Ant。
- Build ScansGradle。

Gradle Gradle。

Gradle 。 JavaGradle。

Examples

Gradle

Java JDKJREGradle 3.x7

1. Gradle
2. ZIP
3. `GRADLE_HOME`。 。
4. `GRADLE_HOME/bin`PATHCLIGradle
5. CLI`gradle -v`Gradle。 GradleGradle

OS X / macOS

gradle

```
brew install gradle
```

SdkMan

SdkManGradle

```
sdk install gradle
```

```
sdk list gradle
sdk install gradle 2.14
```

```
sdk use gradle 2.12
```

EclipseGradle

EclipseGradle

1. Eclipse**Help** - > **Eclipse Marketplace**
2. **buildship**Enter
3. **“Buildship Gradle Integration 1.0” “ ”**
4. “ ”
5. “ ”
6. Eclipse**Yes**

build.gradleGroovyGradle。`> gradle [taskname]`EclipseEclipse。

gradleHello WorldGroovy。 Groovy`printLn`Java`System.out.printLn`。

build.gradle

```
task hello {
    doLast {
        println 'Hello world!'
    }
}
```

`> gradle hello`> gradle -q hello。 `-q`gradle。

`> gradle -q hello`

```
> gradle -q hello
Hello world!
```

operator `<<` leftShift`doLast {closure}`。 **gradle 3.2** 。 **build.gradle**。

- 。 JARJavadoc。

Gradle。

```
task hello {
    doLast{
       //some code
    }
}
```

```
task(hello) {
    doLast{
       //some code
    }
}
```

◦ dependsOn mustRunAfter type◦

```
task hello {
    doLast{
       println 'Inside task'
    }
}
hello.doLast {
    println 'added code'
}
```

```
> gradle -q hello
    Inside task
    added code
```

◦

```
    task hello {
        doLast{
        println "Hello from a simple task"
        }
    }
```

◦ ◦

```
task hello(type: HelloTask)

class HelloTask extends DefaultTask {
    @TaskAction
    def greet() {
        println 'hello from our custom task'
    }
}
```

```
class HelloTask extends DefaultTask {
    String greeting = "This is default greeting"
    @TaskAction
    def greet() {
        println greeting
    }
}
```

```
   //this is our old task definition style
task oldHello(type: HelloTask)
   //this is our new task definition style
task newHello(type: HelloTask) {
    greeting = 'This is not default greeting!'
}
```

```
> gradle -q oldHello
```

```
This is default greeting

> gradle -q newHello
This is not default greeting!
```

gradle https://riptutorial.com/zh-TW/gradle/topic/894/gradle

2: Gradle Init

Examples

init.gradlegradle。 init.gradle。

```
Unix: ~/.gradle/init.gradle
```

init -

- **USER_HOME / .gradle / init.d* .gradle**
- Gradle**init.d* .gradle**

mavenLocalinit.gradle。

```
allprojects {
    repositories {
        mavenLocal()
    }
}
```

maven。“gradle install”jarmavenLocalbuild.gradlenexus / artifactory。

Gradle Init https://riptutorial.com/zh-TW/gradle/topic/4234/gradle-init

3: Gradle Wrapper

Examples

Gradle WrapperGit

gradlejar`gradlew`。 `gradlew`

```
Error: Could not find or load main class org.gradle.wrapper.GradleWrapperMain
```

.gitignoreJava`*jar`。 gradle`gradle/wrapper/gradle-wrapper.jar`。 git。

```
git add -f gradle/wrapper/gradle-wrapper.jar
git ci
```

`-f`。

Gradle Wrapper

Gradle。 Gradle。 。 gradle。

```
> ./gradlew <task> # on *Nix or MacOSX
> gradlew <task>   # on Windows
```

1.

```
gradle wrapper [--gradle-version 2.0]
```

`--gradle-version X`gradle。

1. `build.gradle`

```
task wrapper(type: Wrapper) {
    gradleVersion = '2.0'
}
```

`gradle wrapper`

```
the_project/
  gradlew
  gradlew.bat
  gradle/wrapper/
    gradle-wrapper.jar
    gradle-wrapper.properties
```

https://docs.gradle.org/current/userguide/gradle_wrapper.html 。

Gradle WrapperGradle

GradleWrapper`wrapperdistributionUrl`

```
task wrapper(type: Wrapper) {
    gradleVersion = '2.0'
    distributionUrl = "http\://server/dadada/gradle-${gradleVersion}-bin.zip"
}
```

`gradle wrapper` shell`gradlew` `gradle/wrapper/gradle-wrapper.properties`URLGradle。

Gradle Wrapper

`gradlew`

1. gradle/ .gradle / wrapper / dists
2.

。*JVM*。

`JAVA_OPTSGRADLE_OPTS`

```
-Dhttps.proxyPort=<proxy_port> -Dhttps.proxyHost=<hostname>
```

Windows

```
set JAVA_OPTS=-Dhttps.proxyPort=8080 -Dhttps.proxyHost=myproxy.mycompany.com
```

https://docs.oracle.com/javase/8/docs/api/java/net/doc-files/net-properties。 HTML 。

。

Gradle Wrapper https://riptutorial.com/zh-TW/gradle/topic/3006/gradle-wrapper

4: Gradle

Examples

Gradle。 Gradle`--profile`

```
gradle --profile
./gradlew --profile
```

`./build/reports/profile/./build/reports/profile/`HTML

Profile report

Profiled build: build

Started on: 2016/07/23 - 17:47:33

Summary | Configuration | Depend

Description	Duration
Total Build Time	20.654s
Startup	0.598s
Settings and BuildSrc	0.001s
Loading Projects	0.003s
Configuring Projects	0.061s
Task Execution	19.611s

Generated by Gradle 2.14.1 at Jul 23, 2016 5:47:53 PM

org.gradle.configureondemand“”。

```
org.gradle.configureondemand=true
```

gradle.properties。

“”Gradle。

Gralde

build.gradle。 。 Gradle。

GradleJVM

$GRADLE_USER_HOME/.gradle/gradle.properties ~/.gradle/gradle.properties org.gradle.jvmargsGradle GradleJVM org.gradle.jvmargs 。

gradle.properties。

GradleGradle

```
org.gradle.jvmargs=-Xmx1024m -XX:MaxPermSize=256m
```

1GB“”256MB。 。

```
org.gradle.jvmargs=-Xmx2024m -XX:MaxPermSize=512m
```

XmxXX:MaxPermSize。

Gradle Daemon

Gradle Daemon。

GradleGradle Framework。

--daemongradleGradle Wrapper。

```
gradle --daemon
./gradlew --daemon
```

```
org.gradle.daemon=true
```

gradle.properties。

Gradle$GRADLE_USER_HOME/.gradle/gradle.properties ~/.gradle/gradle.properties

```
org.gradle.daemon=true
```

Mac / Linux / * nix

```
touch ~/.gradle/gradle.properties && echo "org.gradle.daemon=true" >>
~/.gradle/gradle.properties
```

Windows

```
(if not exist "%USERPROFILE%/.gradle" mkdir "%USERPROFILE%/.gradle") && (echo
org.gradle.daemon=true >> "%USERPROFILE%/.gradle/gradle.properties")
```

`--no-daemon--no-daemon gradle.propertiesorg.gradle.daemon=falsegradle.properties。`

`gradle --stop。 --stop`GradleDaemonGradle。 * *3*。

Gradle Parallel

Gradle。`--parallel`Gradle - - 。

```
gradle build --parallel
```

gradle.properties

```
org.gradle.parallel=true
```

Gradle

GradleGradle。 Gradle。 Gradle。 Gradle。

Gradle`gradle/wrapper/gradle-wrapper.properties` 。

```
distributionBase=GRADLE_USER_HOME
distributionPath=wrapper/dists
zipStoreBase=GRADLE_USER_HOME
zipStorePath=wrapper/dists
distributionUrl=https\://services.gradle.org/distributions/gradle-X.X.X.zip
```

`XXX YYY` 。

Gradle https://riptutorial.com/zh-TW/gradle/topic/3443/gradle

5: Gradle

Examples

`buildSrc`gradle

gradleDSL。

。

-
- buildSrc
-

buildSrc。

```
// project's build.gradle
build.gradle
// build.gradle to build the `buildSrc` module
buildSrc/build.gradle
// file name will be the plugin name used in the `apply plugin: $name`
// where name would be `sample` in this example
buildSrc/src/main/resources/META-INF/gradle-plugins/sample.properties
// our DSL (Domain Specific Language) model
buildSrc/src/main/groovy/so/docs/gradle/plugin/SampleModel.groovy
// our actual plugin that will read the values from the DSL
buildSrc/src/main/groovy/so/docs/gradle/plugin/SamplePlugin.groovy
```

build.gradle

```
group 'so.docs.gradle'
version '1.0-SNAPSHOT'

apply plugin: 'groovy'
// apply our plugin... calls SamplePlugin#apply(Project)
apply plugin: 'sample'

repositories {
    mavenCentral()
}

dependencies {
    compile localGroovy()
}

// caller populates the extension model applied above
sample {
    product = 'abc'
    customer = 'zyx'
}

// dummy task to limit console output for example
task doNothing <<{}
```

buildSrc /build.gradle

```
apply plugin: 'groovy'

repositories {
    mavenCentral()
}

dependencies {
    compile localGroovy()
}
```

buildSrc / SRC /// SO // gradle// SamplePlugin.groovy

```
package so.docs.gradle.plugin

import org.gradle.api.Plugin
import org.gradle.api.Project

class SamplePlugin implements Plugin<Project> {
    @Override
    void apply(Project target) {
        // create our extension on the project for our model
        target.extensions.create('sample', SampleModel)
        // once the script has been evaluated the values are available
        target.afterEvaluate {
            // here we can do whatever we need to with our values
            println "populated model: $target.extensions.sample"
        }
    }
}
```

buildSrc / SRC /// SO // gradle// SampleModel.groovy

```
package so.docs.gradle.plugin

// define our DSL model
class SampleModel {
    public String product;
    public String customer;

    @Override
    public String toString() {
        final StringBuilder sb = new StringBuilder("SampleModel{");
        sb.append("product='").append(product).append('\'');
        sb.append(", customer='").append(customer).append('\'');
        sb.append('}');
        return sb.toString();
    }
}
```

buildSrc / src/// META-INF / gradle-/ sample.properties

```
implementation-class=so.docs.gradle.plugin.SamplePlugin
```

DSL

```
 $ ./gradlew -q doNothing
SampleModel{product='abc', customer='zyx'}
```

javaGradleGroovy

```
plugin
|-- build.gradle
|-- settings.gradle
|-- src
    |-- main
    |   |-- java
    |   |-- resources
    |       |-- META-INF
    |           |-- gradle-plugins
    |-- test
```

gradle

build.gradle◦

```
apply plugin: 'java'
apply plugin: 'maven'

dependencies {
    compile gradleApi()
}
```

javajava◦

gradleApi()Gradle◦

settings.gradle

```
rootProject.name = 'myplugin'
```

Maven**ID** ◦

settings.gradle◦

Pluginsrc/main/java/org/sample/MyPlugin.java◦

```
import org.gradle.api.Plugin;
import org.gradle.api.Project;

public class MyPlugin implements Plugin<Project> {

    @Override
    public void apply(Project project) {
         project.getTasks().create("myTask", MyTask.class);
    }

}
```

`DefaultTask`

```
import org.gradle.api.DefaultTask;
import org.gradle.api.tasks.TaskAction;

public class MyTask extends DefaultTask {

    @TaskAction
    public void myTask() {
        System.out.println("Hello World");
    }
}
```

`META-INF/gradle-pluginsimplementation-class`。

`META-INF/gradle-plugins/testplugin.properties`

```
implementation-class=org.sample.MyPlugin.java
```

ID 。

`build.gradle`maven repo

```
apply plugin: 'java'
apply plugin: 'maven'

dependencies {
    compile gradleApi()
}

repositories {
    jcenter()
}


group = 'org.sample'
version = '1.0'

uploadArchives {
    repositories {
        mavenDeployer {
        repository(url: mavenLocal().url)
        }
    }
}
```

Gradle`plugin/build.gradle`Maven。

```
$ ./gradlew clean uploadArchives
```

`build.gradle`

```
buildscript {
     repositories {
         mavenLocal()
     }
 dependencies {
    classpath group: 'org.sample',    // Defined in the build.gradle of the plugin
              name: 'myplugin',       // Defined by the rootProject.name
              version: '1.0'
    }
 }

apply plugin: 'testplugin'            // Defined by the properties filename
```

```
 $ ./gradlew myTask
```

Gradle https://riptutorial.com/zh-TW/gradle/topic/1900/gradle

6: IntelliJ IDEA

- groovy.util.Node = node.find {childNode - > return true ||
- node.appendnodeYouWantAsAChild
- groovy.util.Node parsedNode =new XmlParser。parseTextsomeRawXMLString
- '''mutli-line stringnot interpolated'''

IntelliJ - ipriwsiml - gradle

```
project.ipr
module.iml
workspace.iws
```

.withXmlxml。.asNodegroovy xml。

```
project.ipr.withXml { provider ->
    def node = provider.asNode()
```

- gradleIntelliJIntelliJ。XML。xml。

gradleIntelliJ。。

find==。.contains。

null。

Examples

- `foo.bar.Baz`。
- main。
- `fooBar`。

gradle

```
idea {
    workspace.iws.withXml { provider ->
        // I'm not actually sure why this is necessary
        def node = provider.asNode()

        def runManager = node.find { it.@name.contains('RunManager')}

        // find a run configuration if it' there already
        def runner = runManager.find { it.find ({ mainClass ->
            return mainClass.@name != null && mainClass.@name == "MAIN_CLASS_NAME" &&
mainClass.@value != null && mainClass.@value.contains('Baz');
        }) != null }

        // create and append the run configuration if it doesn't already exists
        if (runManager != null && runner == null){
            def runnerText = '''
```

```
                <configuration default="false" name="Baz" type="Application"
factoryName="Application" nameIsGenerated="true">
                  <extension name="coverage" enabled="false" merge="false" runner="idea">
                    <pattern>
                      <option name="PATTERN" value="foo.bar.Baz" />
                      <option name="ENABLED" value="true" />
                    </pattern>
                  </extension>
                  <option name="MAIN_CLASS_NAME" value="foo.bar.Baz" />
                  <option name="VM_PARAMETERS" value="" />
                  <option name="PROGRAM_PARAMETERS" value="" />
                  <option name="WORKING_DIRECTORY" value="file://$PROJECT_DIR$" />
                  <option name="ALTERNATIVE_JRE_PATH_ENABLED" value="false" />
                  <option name="ALTERNATIVE_JRE_PATH" />
                  <option name="ENABLE_SWING_INSPECTOR" value="false" />
                  <option name="ENV_VARIABLES" />
                  <option name="PASS_PARENT_ENVS" value="true" />
                  <module name="foobar" />
                  <envs />
                  <method />
                </configuration>'''
            runner = (new XmlParser()).parseText(runnerText)
            runManager.append(config);
        }

        // If there is no active run configuration, set the newly made one to be it
        if (runManager != null && runManager.@selected == null) {
            runManager.@selected="${runner.@factoryName}.${runner.@name}"
        }
    }
}
```

IntelliJ IDEA https://riptutorial.com/zh-TW/gradle/topic/2297/intellij-idea

7:

doLast

gradle 3.x **doLast {closure}**“leftShift”<<。 **leftShift**gradle 3.2gradle 5.0。

```
task oldStyle << {
    println 'Deprecated style task'
 }
```

```
task newStyle {
    doLast {
    println 'Deprecated style task'
    }
 }
```

Examples

dependsOn。

```
task A << {
    println 'Hello from A'
}
task B(dependsOn: A) << {
    println "Hello from B"
}
```

`dependsOn

- BA.
- GradleBA。

```
> gradle -q B
Hello from A
Hello from B
```

```
project('projectA') {
    task A(dependsOn: ':projectB:B') << {
        println 'Hello from A'
    }
}

project('projectB') {
    task B << {
        println 'Hello from B'
    }
}
```

:projectB:B:projectB:B 。

```
> gradle -q B
Hello from A
Hello from B
```

```
task A << {
    println 'Hello from A'
}

task B << {
    println 'Hello from B'
}

B.dependsOn A
```

。

```
> gradle -q B
Hello from A
Hello from B
```

。

```
task A << {
    println 'Hello from A'
}

task B << {
    println 'Hello from B'
}

task C << {
    println 'Hello from C'
}

task D << {
    println 'Hello from D'
}
```

```
B.dependsOn A
C.dependsOn B
D.dependsOn C
```

```
> gradle -q D
Hello from A
Hello from B
Hello from C
Hello from D
```

```
B.dependsOn A
D.dependsOn B
D.dependsOn C
```

```
> gradle -q D
```

```
Hello from A
Hello from B
Hello from C
Hello from D
```

dependsOn

。

```
task A << {
    println 'Hello from A'
}

task B(dependsOn: A) << {
    println 'Hello from B'
}

task C << {
    println 'Hello from C'
}

task D(dependsOn: ['B', 'C'] << {
    println 'Hello from D'
}
```

```
> gradle -q D
Hello from A
Hello from B
Hello from C
Hello from D
```

https://riptutorial.com/zh-TW/gradle/topic/5545/

8: GradleAndroid

Examples

```
gradle.taskGraph.whenReady {taskGraph ->
    if (taskGraph.hasTask(assembleDebug)) {  /* when run debug task */
        autoIncrementBuildNumber()
    } else if (taskGraph.hasTask(assembleRelease)) { /* when run release task */
        autoIncrementBuildNumber()
    }
}
```

```
 /*Wrapping inside a method avoids auto incrementing on every gradle task run. Now it runs
only when we build apk*/
ext.autoIncrementBuildNumber = {

    if (versionPropsFile.canRead()) {
        def Properties versionProps = new Properties()
        versionProps.load(new FileInputStream(versionPropsFile))
        versionBuild = versionProps['VERSION_BUILD'].toInteger() + 1
        versionProps['VERSION_BUILD'] = versionBuild.toString()
        versionProps.store(versionPropsFile.newWriter(), null)
    } else {
        throw new GradleException("Could not read version.properties!")
    }
}
```

def versionPropsFile = file'version.properties'def versionBuild

```
/*Setting default value for versionBuild which is the last incremented value stored in the
file */
if (versionPropsFile.canRead()) {
    def Properties versionProps = new Properties()
    versionProps.load(new FileInputStream(versionPropsFile))
    versionBuild = versionProps['VERSION_BUILD'].toInteger()
} else {
    throw new GradleException("Could not read version.properties!")
}
```

GradleAndroid https://riptutorial.com/zh-TW/gradle/topic/10696/gradleandroid

9:

Examples

build.gradle

Gradle *gradle*

build.gradlebuildscript。

```
buildscript {
  repositories {
    maven {
      url "https://plugins.gradle.org/m2/"
    }
  }
  dependencies {
    classpath "org.example.plugin:plugin:1.1.0"
  }
}

apply plugin: "org.example.plugin"
```

Gradle2.1+ *Gradle 2.1*。

```
plugins {
  id "org.example.plugin" version "1.1.0"
}
```

build.gradle

Gradle

buildcriptAllplugin2.1+。

```
buildscript {
  repositories {
    maven {
      url "https://plugins.gradle.org/m2/"
    }
  }
  dependencies {
    classpath "org.example.plugin:plugin:1.1.0"
    Classpath "com.example.plugin2:plugin2:1.5.2"
  }
}

apply plugin: "org.example.plugin"
apply plugin: "com.example.plugin2"
```

Gradle2.1+

```
plugins {
  id "org.example.plugin" version "1.1.0"
  id "com.example.plugin2" version "1.5.2"
}
```

https://riptutorial.com/zh-TW/gradle/topic/9183/

10:

Examples

JAR

JAR

JARGradle。

```
dependencies {
    compile files('path/local_dependency.jar')
}
```

`path local_dependency.jar`JAR。 `path`。

JAR

jar。

```
dependencies {
        compile fileTree(dir: 'libs', include: '*.jar')
}
```

`libs`jar `*.jar`。

JAR

jarflatDir。

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

*`libs`*jar。

GradleMaven。

```
group:name:version
```

```
'org.springframework:spring-core:4.3.1.RELEASE'
```

Gradle`dependency`

```
compile 'org.springframework:spring-core:4.3.1.RELEASE'
```

```
compile group: 'org.springframework', name: 'spring-core', version: '4.3.1.RELEASE'
```

◦

◦

```
testCompile group: 'junit', name: 'junit', version: '4.+'
```

Gradle

gradle◦

```
dependencies {
    compile project(':OtherProject')
}
```

`':OtherProject'`gradle◦

`build.gradle':OtherProject'settings.gradle`

```
include ':Dependency'
project(':Dependency').projectDir = new File('/path/to/dependency')
```

◦

`dependencies`

```
gradle dependencies
```

◦ `--configuration`

```
gradle dependencies --configuration compile
```

`<subproject>:dependencies` task◦ `api`

```
gradle api:dependencies
```

GradleGradle◦ `build.gradlerepositories { ... }build.gradle` ◦

JCenter MavenMaven◦

```
repositories {
  // Adding these two repositories via method calls is made possible by Gradle's Java plugin
  jcenter()
  mavenCentral()

  maven { url "http://repository.of/dependency" }
}
```

gradle.aarAndroid

1. applibs◦
2. .aar.aar◦ myLib.aar ◦
3. app levelbuild.gradleandroid◦

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

applibs◦

4. dependenciesbuild.gradle

```
compile(name:'myLib', ext:'aar')
```

https://riptutorial.com/zh-TW/gradle/topic/2524/

11: Gradle

- - ◦ inputs outputs◦
- dependencies {} - File◦ org.slf4j:slf4j-api:1.7.21Maven ◦
- repositories {} - Gradle◦ ◦ jcenter()maven { url 'http://jcenter.bintray.com/' } } jcenter() : jcenter() maven { url 'http://jcenter.bintray.com/' } }Bintray Maven ◦

Examples

Java

Gradle

Gradle

```
cd $PROJECT_DIR
gradle init --type=java-library
```

ScalaJava◦

```
.
├── build.gradle
├── gradle
│   └── wrapper
│       ├── gradle-wrapper.jar
│       └── gradle-wrapper.properties
├── gradlew
├── gradlew.bat
├── settings.gradle
└── src
    ├── main
    │   └── java
    │       └── Library.java
    └── test
        └── java
            └── LibraryTest.java
```

gradle tasks jar test javadocbuild.gradle

```
apply plugin: 'java'

repositories {
    jcenter()
}

dependencies {
    compile 'org.slf4j:slf4j-api:1.7.21'
    testCompile 'junit:junit:4.12'
}
```

Gradle https://riptutorial.com/zh-TW/gradle/topic/2247/gradle

12: -

model.android.ndk.toolchain	ndk-bundle

Examples

JNI Gradle

rootbuild.gradle

```
buildscript {
    repositories {
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle-experimental:0.8.0-alpha4'
    }
}

allprojects {
    repositories {
        jcenter()
    }
}
```

appbuild.gradle

```
apply plugin: 'com.android.model.application'

dependencies {
    compile "com.android.support:support-v4:23.3.0"
    compile fileTree(dir: 'libs', include: '*.jar')
}

model {
    android {
        compileSdkVersion = 23
        buildToolsVersion = '23.0.3'

        defaultConfig {
            applicationId = 'com.example.hello'
            minSdkVersion.apiLevel = 9
            targetSdkVersion.apiLevel = 23

            buildConfigFields {
                create() {
                    type "int"
                    name "VALUE"
                    value "1"
                }
            }
        }
```

```
        ndk {
            platformVersion = 9
            moduleName "hello"

            toolchain "clang"

            stl "gnustl_static"
            CFlags.add("-DANDROID_NDK")
            cppFlags.add("-std=c++11")

            ldLibs.add("android")
            ldLibs.add("dl")
            ldLibs.add("log")
        }

        sources {
            main {
                jni {
                    exportedHeaders {
                        srcDirs "../../common/headers"
                    }
                    source {
                        srcDirs "../../common/src"
                    }
                }
            }
        }
    }
}
```

OpenGL ES 2.0

rootbuild.gradle

```
buildscript {
    repositories {
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle-experimental:0.8.0-alpha4'
    }
}

allprojects {
    repositories {
        jcenter()
    }
}
```

appbuild.gradle

```
apply plugin: 'com.android.model.application'

dependencies {
    compile "com.android.support:support-v4:23.3.0"
    compile fileTree(dir: 'libs', include: '*.jar')
}
```

```
model {
    android {
        compileSdkVersion = 23
        buildToolsVersion = '23.0.3'

        defaultConfig {
            applicationId = 'com.example.glworld'
            minSdkVersion.apiLevel = 9
            targetSdkVersion.apiLevel = 23

            buildConfigFields {
                create() {
                    type "int"
                    name "VALUE"
                    value "1"
                }
            }
        }

        buildTypes {
            release {
                minifyEnabled = false
                proguardFiles.add(file('proguard-rules.txt'))
            }
        }

        ndk {
            platformVersion = 9
            moduleName "glworld"

            toolchain "clang"

            stl "gnustl_static"
            CFlags.add("-DANDROID_NDK")
            CFlags.add("-DDISABLE_IMPORTGL")
            CFlags.add("-DFT2_BUILD_LIBRARY=1")
            cppFlags.add("-std=c++11")

            ldLibs.add("EGL")
            ldLibs.add("android")
            ldLibs.add("GLESv2")
            ldLibs.add("dl")
            ldLibs.add("log")
        }

        sources {
            main {
                jni {
                    dependencies {
                        library "freetype2" linkage "shared"
                    }
                    exportedHeaders {
                        srcDirs "../../common/headers"
                    }
                    source {
                        srcDirs "../../common/src"
                    }
                }
            }
        }
```

```
    }

    repositories {
        prebuilt(PrebuiltLibraries) {
            freetype2 {
                headers.srcDir "../../common/freetype2-android/include"
                binaries.withType(SharedLibraryBinary) {
                    def localLib = "../../common/freetype2-android/Android/libs"
                    sharedLibraryFile =
                            file("$localLib/${targetPlatform.getName()}/libfreetype2.so")
                }
            }
        }
    }
}

// The next tasks compile a freetype library using a make file.
// These `.so`'s are then used as the shared libraries compiled above.
tasks.withType(JavaCompile) {
    compileTask -> compileTask.dependsOn buildNative
}

// Call regular ndk-build (.cmd) script from the app directory
task buildNative(type: Exec) {
    def ndkDir = "/Development/android-sdk-macosx/ndk-bundle"
    commandLine "$ndkDir/ndk-build",
            '-C',
            file('../../common/freetype2-android/Android/jni').absolutePath
}

task cleanNative(type: Exec) {
    def ndkDir = "/Development/android-sdk-macosx/ndk-bundle"
    commandLine "$ndkDir/ndk-build",
            '-C',
            file('../../common/freetype2-android/Android/jni').absolutePath,
            "clean"
}

clean.dependsOn cleanNative
```

- https://riptutorial.com/zh-TW/gradle/topic/4460/---

13:

`mustRunAftershouldRunAfter`""Gradle 3.0。

- `mustRunAfter`
- `shouldRunAfter`

`mustRunAfter`taskAtaskBtaskBtaskA。

`shouldRunAfter`

- 。
- `shouldRunAftershouldRunAfter`。

Examples

mustRunAfter

```
task A << {
    println 'Hello from A'
}
task B << {
    println 'Hello from B'
}

B.mustRunAfter A
```

`B.mustRunAfter A`Gradle。

```
> gradle -q B A
Hello from A
Hello from B
```

AB。

AB.

```
> gradle -q B
Hello from B
```

https://riptutorial.com/zh-TW/gradle/topic/5550/

S. No		Contributors
1	gradle	Afterfield, bassim, Community, Emil Burzo, Eric Wendelin, Hamzawey, Hillkorn, Matthias Braun, Nikem, Pepper Lebeck-Jobe, Sergey Yakovlev, Stanislav, user2555595, vanogrid, Will
2	Gradle Init	ambes, Hillkorn
3	Gradle Wrapper	ajoberstar, Fanick, HankCa, I Stevenson
4	Gradle	ambes, Sergey Yakovlev, Will
5	Gradle	Gabriele Mariotti, JBirdVegas
6	IntelliJ IDEA	IronHorse, Sam Sieber, Will
7		Gabriele Mariotti, Sergey Yakovlev, Stanislav
8	GradleAndroid	Jayakrishnan PM
9		Afterfield
10		Afshin, Andrii Abramov, GameScripting, Hillkorn, leeor, Matthias Braun, mcarlin, mszymborski, Will
11	Gradle	Eric Wendelin, Will
12	-	iHowell
13		Gabriele Mariotti