Gradle

一，特点

1，基于合约的灵活的通用构建工具

2，支持已有的 maven 和 ivy 仓库

3，支持传递性依赖管理, 而不需要远程仓库或者 pom.xml 或者 ivy 配置文件

4，使用构建缓存、守护进程等方式提高编译速度

二，多项目的 Java 构建

*0 ,*构建布局

multiproject/

api/

services/webservice/

shared/

settings.gradle

build.gradle

*1,*设置文件*.* 放在源代码的根目录*,* 它指定要包含哪个项目*.* 它的名字必须叫做***settings.gradle****.*

*settings.gradle*

include "shared", "api", "services:webservice", "services:shared"

*2,*通用配置*.*放在源代码的根目录，指定所有子项目的通用配置

build.gradle

subprojects { //subprojects 方法遍历根项目的所有元素并且注入指定的配置

apply plugin: ‘java' //gradle的Java 插件。集成了大多数java项目的构建任务。可以使用 gradle tasks 来列出项目的所有任务

apply plugin: 'eclipse-wtp'

repositories { //指定外部依赖。加入 Maven 仓库

mavenCentral() //Maven central 仓库

maven { //远程的 Maven 仓库

url "http://repo.mycompany.com/maven2"

}

}

dependencies { //指定外部依赖。加入依赖

compile group: 'commons-collections', name: 'commons-collections', version: ‘3.2' //编译阶段的依赖。此处是依赖的标准写法

compile group: 'org.hibernate', name: 'hibernate-core', version: '3.6.7.Final'

testCompile ‘junit:junit:4.11' //测试和编译阶段的依赖。此处是依赖的精简写法，等效于标准写法

}

version = '1.0'

jar {

manifest.attributes provider: 'gradle'

}

}

*3*，项目之间的依赖

若在 *api* 构建文件里我们将加入一个由 *shared* 项目产生的 *JAR* 文件的依赖，

则

*api/build.gradle*

dependencies {

compile project(':shared')

}

*4*，创建一个发行版本

api/build.gradle

task dist(type: Zip) {

dependsOn spiJar

from 'src/dist'

into('libs') {

from spiJar.archivePath

from configurations.runtime

}

}

artifacts {

archives dist

}

三，增量构建

[up-to-date](https://docs.gradle.org/current/userguide/more_about_tasks.html#sec:up_to_date_checks)的概念，可以自动跳过没有任何更新的 Task 从而加快构建速度。使用 TaskInputs 和 TaskOutputs 属性定义好 Task 的输入输出文件之后，在第一次运行 Task 的时候，Gradle 会记录 input 文件内容的 Hash 值快照，也会记录下 Task 运行成功之后的 output 快照。而在这之后，每当 Task 被执行之前 Gradle 就会以前所保存的快照进行对比，只有在有差异的情况才会重新执行该 Task。

apply plugin: 'java'

compileJava {

//enable compilation in a separate daemon process

options.fork = true

//enable incremental compilation

options.incremental = true

}

四，一个spring boot项目的build.gradle样例

allprojects {

group 'com.chen'

version '1.0-SNAPSHOT'

repositories {

maven { url 'http://maven.aliyun.com/nexus/content/groups/public/' }

}

}

buildscript {

ext {

springIOVersion = '1.0.0.RELEASE'

springBootVersion = '1.5.9.RELEASE'

}

repositories {

jcenter()

mavenLocal()

mavenCentral()

maven { url "http://repo.spring.io/release" }

maven { url "http://repo.spring.io/milestone" }

maven { url "http://repo.spring.io/snapshot" }

maven { url "https://plugins.gradle.org/m2/" }

}

dependencies {

classpath "io.spring.gradle:dependency-management-plugin:${springIOVersion}"

classpath "org.springframework.boot:spring-boot-gradle-plugin:${springBootVersion}"

}

}

tasks.withType(JavaCompile) {

options.encoding = "UTF-8"

}

apply plugin: 'idea'

apply plugin: 'java'

apply plugin: 'spring-boot'

apply plugin: 'io.spring.dependency-management'

sourceCompatibility = 1.8

targetCompatibility = 1.8

jar {

baseName = 'gradle-demo'

version = '0.0.1'

manifest {

attributes "Manifest-Version": 1.0,

'Main-Class': 'com.chen.GradleDemoApplication'

}

}

repositories {

jcenter()

mavenLocal()

mavenCentral()

}

dependencyManagement {

imports {

mavenBom 'io.spring.platform:platform-bom:Brussels-SR6'

mavenBom 'org.springframework.cloud:spring-cloud-dependencies:Brixton.SR4'

}

}

ext {

junitVersion = '4.12'

}

dependencies {

compile 'org.springframework:spring-core'

compile 'org.springframework.boot:spring-boot-starter-web'

compile 'org.springframework.boot:spring-boot-autoconfigure'

compile 'org.springframework.boot:spring-boot-starter-tomcat'

testCompile 'org.springframework.boot:spring-boot-starter-test'

testCompile "junit:junit:${junitVersion}"

}

心得：

gradle作为一个新兴的工具已经有了广泛的应用。spring boot等项目已经从Maven切换到了gradle。开发安卓程序也只支持gradle了。因此不管是否现在需要将项目从maven切换到gradle，但是至少学习gradle是一件必要的事情。

参考

<https://mp.weixin.qq.com/s/tkGfeRNM4Hnzinzxl8-DKQ>

<https://blog.jimmylv.info/2015-11-16-how-to-use-gradle-and-speed-up-build-time/>

<https://wiki.jikexueyuan.com/project/GradleUserGuide-Wiki/>

<https://www.jianshu.com/p/9231b1f598c5>