## 常用命令

gradle project

gradle clean

gradle build

gradle tasks

gradle properties

## 依赖管理与胖产品包

|  |
| --- |
| apply plugin: 'java'  */\*\*  \* 打包文件名  \*/* archivesBaseName = "ifollowyou"  */\*\*  \* 指定编译编码  \*/* [compileJava,compileTestJava,javadoc]\*.options\*.encoding = "UTF-8"  repositories {  maven {  url 'http://localhost:8081/nexus/content/groups/public/'  }   mavenLocal()  jcenter() }  dependencies {  compile 'ch.qos.logback:logback-classic:1.1.3'  testCompile 'junit:junit:4.12' }  jar {  manifest {  attributes 'Main-Class': 'net.petrikainulainen.gradle.HelloWorld'  }   from { configurations.compile.collect { it.isDirectory() ? it : zipTree(it) } } } |

## 二进制产品包

|  |
| --- |
| **apply plugin: 'application'** apply plugin: 'java'  mainClassName = 'net.petrikainulainen.gradle.HelloWorld'  repositories {  maven {  url 'http://localhost:8081/nexus/content/groups/public/'  }   mavenLocal()  jcenter() }  dependencies {  compile 'ch.qos.logback:logback-classic:1.1.3'  testCompile 'junit:junit:4.12' }  task copyLicense {  outputs.file new File("$buildDir/LICENSE")   doLast {  copy {  from "LICENSE"  into "$buildDir"  }  } }  applicationDistribution.from(copyLicense) {  into "" } |

## WEB应用

|  |
| --- |
| buildscript {  repositories {  maven {  url 'http://127.0.0.1:8081/nexus/content/groups/public/'  }  }  dependencies {  classpath 'org.akhikhl.gretty:gretty:1.2.4'  } }  apply plugin: 'java' apply plugin: 'war' apply plugin: 'org.akhikhl.gretty'  gretty {  port = 8080  contextPath = '/'  servletContainer = 'jetty9' }  archivesBaseName = 'ifollowyou' sourceCompatibility = '1.8' targetCompatibility = '1.8'  [compileJava,compileTestJava,javadoc]\*.options\*.encoding = "UTF-8"  repositories {  maven {  url 'http://127.0.0.1:8081/nexus/content/groups/public/'  }   mavenLocal()  jcenter() }  dependencies {  compile(  'org.springframework:spring-webmvc:4.1.5.RELEASE',  'org.slf4j:slf4j-api:1.7.10'  )  providedCompile(  'javax.servlet:javax.servlet-api:3.1.0',  )  runtime(  'ch.qos.logback:logback-classic:1.1.3',  'org.slf4j:jcl-over-slf4j:1.7.10',  )  testCompile(  'junit:junit:4.12',  'org.springframework:spring-test:4.1.5.RELEASE'  ) } |

## 多工程项目

### settings.gradle

|  |
| --- |
| include 'app' include 'core' |

### build.gradle

|  |
| --- |
| subprojects {  apply plugin: 'java'  [compileJava,compileTestJava,javadoc]\*.options\*.encoding = "UTF-8"  repositories {  maven {  url 'http://127.0.0.1:8081/nexus/content/groups/public/'  }   mavenLocal()  jcenter()  } } |

### 工程依赖

|  |
| --- |
| apply plugin: 'application'  dependencies {  compile 'org.slf4j:slf4j-api:1.7.12'  compile project(':core') }  mainClassName = 'net.petrikainulainen.gradle.client.HelloWorld'  task copyLicense {  outputs.file new File("$buildDir/LICENSE")  doLast {  copy {  from "LICENSE"  into "$buildDir"  }  } }  applicationDistribution.from(copyLicense) {  into "" } |

## 简单集成测试

|  |
| --- |
| apply plugin: 'application' apply plugin: 'java'  repositories {  maven {  url 'http://localhost:8081/nexus/content/groups/public/'  }   mavenLocal()  jcenter()   mavenCentral() }  [compileJava,compileTestJava,javadoc]\*.options\*.encoding = "UTF-8"  //Configures the classpath and the source and resource directories //of the integrationTest task. sourceSets {  //Declaring this source set creates compile time and runtime configurations for it.  integrationTest {  java {  compileClasspath += main.output + test.output  runtimeClasspath += main.output + test.output  srcDir file('src/integration-test/java')  }  resources.srcDir file('src/integration-test/resources')  } }  //Configures the compile and runtime configurations for our integration tests //and includes the dependencies of the test task. configurations {  integrationTestCompile.extendsFrom testCompile  integrationTestRuntime.extendsFrom testRuntime }  dependencies {  compile 'ch.qos.logback:logback-classic:1.1.3'  testCompile 'junit:junit:4.11'  integrationTestCompile 'org.assertj:assertj-core:3.1.0' }  //Create the task that runs the integration tests found from the //configured source directory and uses the correct classpath. task integrationTest(type: Test) {  testClassesDir = sourceSets.integrationTest.output.classesDir  classpath = sourceSets.integrationTest.runtimeClasspath  //If you want to ensure that integration tests are run every time when you invoke  //this task, uncomment the following line.  //outputs.upToDateWhen { false } }  //Ensure that the check task fails the build if there are failing integration tests. check.dependsOn integrationTest //Ensure that our unit tests are run before our integration tests integrationTest.mustRunAfter test  //Ensure that the HTML reports of unit and integration tests are written to different directories. tasks.withType(Test) {  reports.html.destination = file("${reporting.baseDir}/${name}") }  mainClassName = 'net.petrikainulainen.gradle.HelloWorld'  task copyLicense {  outputs.file new File("$buildDir/LICENSE")  doLast {  copy {  from "LICENSE"  into "$buildDir"  }  } }  applicationDistribution.from(copyLicense) {  into "" } |

## 集成测试插件

|  |
| --- |
| buildscript {  repositories {  maven {  url 'http://localhost:8081/nexus/content/groups/public/'  }   mavenLocal()  jcenter()   }  dependencies {  classpath 'org.unbroken-dome.gradle-plugins:gradle-testsets-plugin:1.2.0'  } }  apply plugin: 'application' apply plugin: 'java' apply plugin: 'org.unbroken-dome.test-sets'  repositories {  maven {  url 'http://localhost:8081/nexus/content/groups/public/'  }   mavenLocal()  mavenCentral() }  testSets {  integrationTest { dirName = 'integration-test' } }  dependencies {  compile 'ch.qos.logback:logback-classic:1.1.3'   testCompile 'junit:junit:4.12'  testCompile 'org.assertj:assertj-core:3.1.0' }  //If you want to ensure that integration tests are run every time when you invoke //this task, uncomment the following block. project.integrationTest {  outputs.upToDateWhen { false } }  //Ensure that the check task fails the build if there are failing integration tests. check.dependsOn integrationTest //Ensure that our unit tests are run before our integration tests integrationTest.mustRunAfter test  //Ensure that the HTML reports of unit and integration tests are written to different directories. tasks.withType(Test) {  reports.html.destination = file("${reporting.baseDir}/${name}") }  mainClassName = 'net.petrikainulainen.gradle.HelloWorld'  task copyLicense {  outputs.file new File("$buildDir/LICENSE")  doLast {  copy {  from "LICENSE"  into "$buildDir"  }  } }  applicationDistribution.from(copyLicense) {  into "" } |

## SpringBoot工程

|  |
| --- |
| buildscript {  repositories {  maven {  url 'http://127.0.0.1:8081/nexus/content/groups/public/'  }  jcenter()  }  dependencies {  classpath(  'org.springframework.boot:spring-boot-gradle-plugin:1.2.5.RELEASE',  'org.unbroken-dome.gradle-plugins:gradle-testsets-plugin:1.2.0'  )  } }  apply plugin: 'java' apply plugin: 'org.unbroken-dome.test-sets' apply plugin: 'spring-boot'  archivesBaseName = 'ifollowyou' sourceCompatibility = 1.8 targetCompatibility = 1.8  repositories {  maven {  url 'http://127.0.0.1:8081/nexus/content/groups/public/'  }   mavenLocal()  jcenter()  mavenCentral() }  testSets {  integrationTest { dirName = 'integration-test' } }  dependencies {  compile(  'org.springframework.boot:spring-boot-starter-actuator',  'org.springframework.boot:spring-boot-starter-thymeleaf'  )  testCompile(  'com.nitorcreations:junit-runners:1.2',  'org.springframework.boot:spring-boot-starter-test'  ) }  //If you want to ensure that integration tests are run every time when you invoke //this task, uncomment the following block. project.integrationTest {  outputs.upToDateWhen { false } }  //Ensure that the check task fails the build if there are failing integration tests. check.dependsOn integrationTest //Ensure that our unit tests are run before our integration tests integrationTest.mustRunAfter test  //Ensure that the HTML reports of unit and integration tests are written to different directories. tasks.withType(Test) {  reports.html.destination = file("${reporting.baseDir}/${name}") } |