Docs

[User Manual](http://docs.google.com/userguide/userguide.html)

[Guides and Tutorials](https://guides.gradle.org)

[DSL Reference](http://docs.google.com/dsl/)

[Javadoc](http://docs.google.com/javadoc/)

[Release Notes](http://docs.google.com/release-notes.html)

[Forums](https://discuss.gradle.org/)

[Training](https://gradle.org/training/)

[Try Gradle Enterprise](https://gradle.com/enterprise)

[PDF](http://docs.google.com/userguide/userguide.pdf)

* [User Manual Home](http://docs.google.com/userguide/userguide.html)
* [Release Notes](http://docs.google.com/release-notes.html)
* [Installing Gradle](http://docs.google.com/userguide/installation.html)
* [Tutorials](https://guides.gradle.org/)

### Reference

* [Groovy DSL Reference](http://docs.google.com/dsl/)
* [Gradle API Javadoc](http://docs.google.com/javadoc/)
* [Core Plugins](http://docs.google.com/userguide/plugin_reference.html)
* [Gradle & Third-party Tools](http://docs.google.com/userguide/third_party_integration.html)

### Getting Started

* [Creating New Gradle Builds](https://guides.gradle.org/creating-new-gradle-builds/)
* [Creating Build Scans](https://guides.gradle.org/creating-build-scans/)
* [Migrating From Maven](https://guides.gradle.org/migrating-from-maven/)

### Running Gradle Builds

* [Command-Line Interface](http://docs.google.com/userguide/command_line_interface.html)
* [Customizing Execution](#gjdgxs)
  + [Configuring the Build Environment](http://docs.google.com/userguide/build_environment.html)
  + [Configuring the Gradle Daemon](http://docs.google.com/userguide/gradle_daemon.html)
  + [Initialization Scripts](http://docs.google.com/userguide/init_scripts.html)
* [Directory Layout](http://docs.google.com/userguide/directory_layout.html)
* [Executing Multi-Project Builds](http://docs.google.com/userguide/intro_multi_project_builds.html)
* [Gradle Wrapper](http://docs.google.com/userguide/gradle_wrapper.html)
* [Troubleshooting](http://docs.google.com/userguide/troubleshooting.html)
* [Using Build Scans](https://docs.gradle.com/build-scan-plugin)
* [Enabling and Configuring the Build Cache](http://docs.google.com/userguide/build_cache.html)
* [Integrating Separate Gradle Builds (Composite Builds)](http://docs.google.com/userguide/composite_builds.html)

### Authoring Gradle Builds

* [Fundamentals](#30j0zll)
  + [Introducing the Basics of Build Scripts](http://docs.google.com/userguide/tutorial_using_tasks.html)
  + [Working with Tasks](http://docs.google.com/userguide/more_about_tasks.html)
  + [Learning More About Build Scripts](http://docs.google.com/userguide/writing_build_scripts.html)
  + [Working with Files](http://docs.google.com/userguide/working_with_files.html)
  + [Creating Custom Task Types](http://docs.google.com/userguide/custom_tasks.html)
  + [Using Gradle Plugins](http://docs.google.com/userguide/plugins.html)
  + [The Standard Gradle Plugins](http://docs.google.com/userguide/standard_plugins.html)
  + [Understanding the Build Lifecycle](http://docs.google.com/userguide/build_lifecycle.html)
  + [Working with Logging](http://docs.google.com/userguide/logging.html)
  + [Configuring Multi-Project Builds](http://docs.google.com/userguide/multi_project_builds.html)
* [Best Practices](#1fob9te)
  + [Authoring Maintainable Build Scripts](http://docs.google.com/userguide/authoring_maintainable_build_scripts.html)
  + [Organizing Gradle Projects](http://docs.google.com/userguide/organizing_gradle_projects.html)
  + [Optimizing Build Performance](https://guides.gradle.org/performance/)
  + [Using the Build Cache](https://guides.gradle.org/using-build-cache/)
* [Dependency Management](#3znysh7)
  + [Introduction to Dependency Management](http://docs.google.com/userguide/introduction_dependency_management.html)
  + [Dependency Management Terminology](http://docs.google.com/userguide/dependency_management_terminology.html)
  + [Dependency Types](http://docs.google.com/userguide/dependency_types.html)
  + [Repository Types](http://docs.google.com/userguide/repository_types.html)
  + [Declaring Dependencies](http://docs.google.com/userguide/declaring_dependencies.html)
  + [Declaring Repositories](http://docs.google.com/userguide/declaring_repositories.html)
  + [Inspecting Dependencies](http://docs.google.com/userguide/inspecting_dependencies.html)
  + [Managing Dependency Configurations](http://docs.google.com/userguide/managing_dependency_configurations.html)
  + [Managing Transitive Dependencies](http://docs.google.com/userguide/managing_transitive_dependencies.html)
  + [Dependency Locking](http://docs.google.com/userguide/dependency_locking.html)
  + [Troubleshooting Dependency Resolution](http://docs.google.com/userguide/troubleshooting_dependency_resolution.html)
  + [Customizing Dependency Resolution Behavior](http://docs.google.com/userguide/customizing_dependency_resolution_behavior.html)
  + [Dependency Cache Internals](http://docs.google.com/userguide/dependency_cache.html)
  + [Working with Dependencies](http://docs.google.com/userguide/working_with_dependencies.html)
* [Publishing Artifacts](http://docs.google.com/userguide/artifact_management.html)
* [C++ Projects](#2et92p0)
  + [Building Native Software](http://docs.google.com/userguide/native_software.html)
  + [Software Model Concepts](http://docs.google.com/userguide/software_model_concepts.html)
  + [Rule-based Model Configuration](http://docs.google.com/userguide/software_model.html)
  + [Implementing Model Rules in a Plugin](http://docs.google.com/userguide/rule_source.html)
  + [Extending the Software Model](http://docs.google.com/userguide/software_model_extend.html)
* [Java Projects](#tyjcwt)
  + [Building Java & JVM projects](http://docs.google.com/userguide/building_java_projects.html)
  + [Testing Java & JVM projects](http://docs.google.com/userguide/java_testing.html)
* [Advanced Techniques](#3dy6vkm)
  + [Configuring Tasks Lazily](http://docs.google.com/userguide/lazy_configuration.html)
  + [Developing Parallel Tasks](https://guides.gradle.org/using-the-worker-api/)
  + [Testing Your Build with TestKit](http://docs.google.com/userguide/test_kit.html)
  + [Using Ant from Gradle](http://docs.google.com/userguide/ant.html)
* [Sample Gradle builds](#1t3h5sf)
  + [Groovy DSL Samples](https://github.com/gradle/gradle/tree/master/subprojects/docs/src/samples)
  + [Kotlin DSL Samples](https://github.com/gradle/kotlin-dsl/tree/master/samples)

### Extending Gradle

* [Writing Custom Plugins](http://docs.google.com/userguide/custom_plugins.html)
* [Plugin Development Guides](https://gradle.org/guides/?q=Plugin+Development)

[Edit this page](https://github.com/gradle/gradle/edit/master/subprojects/docs/src/docs/userguide/)

# Working with Dependencies

Contents

[Iterating over dependencies assigned to a configuration](#4d34og8)

[Iterating over artifacts resolved for a module](#2s8eyo1)

[Navigating the dependency graph](#17dp8vu)

[Accessing a module’s metadata file](#3rdcrjn)

Gradle provides an extensive API for navigating, inspecting and post-processing metadata and artifacts of resolved dependencies.

The main entry point for this functionality is the [Configuration](http://docs.google.com/dsl/org.gradle.api.artifacts.Configuration.html) API. To learn more about the fundamentals of configurations, see [Managing Dependency Configurations](http://docs.google.com/managing_dependency_configurations.html#managing_dependency_configurations).

[Iterating over dependencies assigned to a configuration](#4d34og8)

Sometimes you’ll want to implement logic based on the dependencies declared in the build script of a project e.g. to inspect them in a Gradle plugin. You can iterate over the set of dependencies assigned to a configuration with the help of the method [Configuration.getDependencies()](http://docs.google.com/dsl/org.gradle.api.artifacts.Configuration.html#org.gradle.api.artifacts.Configuration:dependencies). Alternatively, you can also use [Configuration.getAllDependencies()](http://docs.google.com/dsl/org.gradle.api.artifacts.Configuration.html#org.gradle.api.artifacts.Configuration:allDependencies) to include the dependencies declared in [superconfigurations](http://docs.google.com/managing_dependency_configurations.html#sub:inheriting_dependencies_from_other_configurations). These APIs only return the declared dependencies and do not trigger [dependency resolution](http://docs.google.com/introduction_dependency_management.html#sec:dependency_resolution). Therefore, the dependency sets do not include transitive dependencies. Calling the APIs during the [configuration phase of the build lifecycle](http://docs.google.com/build_lifecycle.html#sec:build_phases) does not result in a significant performance impact.

[Example: Iterating over the dependencies assigned to a configuration](#26in1rg)

**build.gradle**

task iterateDeclaredDependencies {  
 doLast {  
 DependencySet dependencySet = configurations.scm.dependencies  
  
 dependencySet.each {  
 logger.quiet "$it.group:$it.name:$it.version"  
 }  
 }  
}

[Iterating over artifacts resolved for a module](#2s8eyo1)

None of the [dependency reporting](http://docs.google.com/inspecting_dependencies.html#inspecting_dependencies) helps you with inspecting or further processing the underlying, resolved artifacts of a module. A typical use case for accessing the artifacts is to copy them into a specific directory or filter out files of interest based on a specific file extension.

You can iterate over the complete set of artifacts resolved for a module with the help of the method [FileCollection.getFiles()](http://docs.google.com/javadoc/org/gradle/api/file/FileCollection.html#getFiles--). Every file instance returned from the method points to its location in the [dependency cache](http://docs.google.com/dependency_cache.html#dependency_cache). Using this method on a Configuration instance is possible as the interface extends FileCollection.

[Example: Iterating over the artifacts resolved for a module](#lnxbz9)

**build.gradle**

task iterateResolvedArtifacts {  
 dependsOn configurations.scm  
  
 doLast {  
 configurations.scm.each {  
 logger.quiet it.absolutePath  
 }  
 }  
}

| **✨** | Iterating over the artifacts of a module automatically resolves the configuration. A resolved configuration becomes immutable and cannot add or remove dependencies. If needed you can copy a configuration for further modification via [Configuration.copy()](http://docs.google.com/dsl/org.gradle.api.artifacts.Configuration.html#org.gradle.api.artifacts.Configuration:copy()). |
| --- | --- |

[Navigating the dependency graph](#17dp8vu)

As a plugin developer, you may want to navigate the full graph of dependencies assigned to a configuration e.g. for turning the dependency graph into a visualization. You can access the full graph of dependencies for a configuration with the help of the [ResolutionResult](http://docs.google.com/javadoc/org/gradle/api/artifacts/result/ResolutionResult.html).

The resolution result provides various methods for accessing the resolved and unresolved dependencies. For demonstration purposes the sample code uses [ResolutionResult.getRoot()](http://docs.google.com/javadoc/org/gradle/api/artifacts/result/ResolutionResult.html#getRoot--) to access the root node the resolved dependency graph. Each dependency of this component returns an instance of [ResolvedDependencyResult](http://docs.google.com/javadoc/org/gradle/api/artifacts/result/ResolvedDependencyResult.html) or [UnresolvedDependencyResult](http://docs.google.com/javadoc/org/gradle/api/artifacts/result/UnresolvedDependencyResult.html) providing detailed information about the node.

[Example: Walking the resolved and unresolved dependencies of a configuration](#35nkun2)

**build.gradle**

task walkDependencyGraph(type: DependencyGraphWalk) {  
 dependsOn configurations.scm  
}  
  
class DependencyGraphWalk extends DefaultTask {  
 @TaskAction  
 void walk() {  
 Configuration configuration = project.configurations.scm  
 ResolutionResult resolutionResult = configuration.incoming.resolutionResult  
 ResolvedComponentResult root = resolutionResult.root  
 logger.quiet configuration.name  
 traverseDependencies(0, root.dependencies)  
 }  
  
 private void traverseDependencies(int level, Set<? extends DependencyResult> results) {  
 for (DependencyResult result : results) {  
 if (result instanceof ResolvedDependencyResult) {  
 ResolvedComponentResult componentResult = result.selected  
 ComponentIdentifier componentIdentifier = componentResult.id  
 String node = calculateIndentation(level) + "- $componentIdentifier.displayName ($componentResult.selectionReason)"  
 logger.quiet node  
 traverseDependencies(level + 1, componentResult.dependencies)  
 } else if (result instanceof UnresolvedDependencyResult) {  
 ComponentSelector componentSelector = result.attempted  
 String node = calculateIndentation(level) + "- $componentSelector.displayName (failed)"  
 logger.quiet node  
 }  
 }  
 }  
  
 private String calculateIndentation(int level) {  
 ' ' \* level  
 }  
}

[Accessing a module’s metadata file](#3rdcrjn)

As part of the dependency resolution process, Gradle downloads the metadata file of a module and stores it in the dependency cache. Some organizations enforce strong restrictions on accessing repositories outside of internal network. Instead of downloading artifacts, those organizations prefer to provide an "installable" Gradle cache with all artifacts contained in it to fulfill the build’s dependency requirements.

The artifact query API provides access to the raw files of a module. Currently, it allows getting a handle to the metadata file and some selected, additional artifacts (e.g. a JVM-based module’s source and Javadoc files). The main API entry point is [ArtifactResolutionQuery](http://docs.google.com/dsl/org.gradle.api.artifacts.query.ArtifactResolutionQuery.html).

Let’s say you wanted to post-process the metadata file of a Maven module. The group, name and version of the module component serve as input to the artifact resolution query. After executing the query, you get a handle to all components that match the criteria and their underlying files. Additionally, it’s very easy to post-process the metadata file. The example code uses Groovy’s [XmlSlurper](http://docs.groovy-lang.org/latest/html/api/groovy/util/XmlSlurper.html) to ask for POM element values.

[Example: Accessing a Maven module’s metadata artifact](#1ksv4uv)

**build.gradle**

apply plugin: 'java-library'  
  
repositories {  
 mavenCentral()  
}  
  
dependencies {  
 implementation 'com.google.guava:guava:18.0'  
}  
  
task printGuavaMetadata {  
 dependsOn configurations.compileClasspath  
  
 doLast {  
 ArtifactResolutionQuery query = dependencies.createArtifactResolutionQuery()  
 .forModule('com.google.guava', 'guava', '18.0')  
 .withArtifacts(MavenModule, MavenPomArtifact)  
 ArtifactResolutionResult result = query.execute()  
  
 for(component in result.resolvedComponents) {  
 Set<ArtifactResult> mavenPomArtifacts = component.getArtifacts(MavenPomArtifact)  
 ArtifactResult guavaPomArtifact = mavenPomArtifacts.find { it.file.name == 'guava-18.0.pom' }  
 def xml = new XmlSlurper().parse(guavaPomArtifact.file)  
 println guavaPomArtifact.file  
 println xml.name  
 println xml.description  
 }  
 }  
}

Docs

* [User Manual](http://docs.google.com/userguide/userguide.html)
* [DSL Reference](http://docs.google.com/dsl/)
* [Release Notes](http://docs.google.com/release-notes.html)
* [Javadoc](http://docs.google.com/javadoc/)

News

* [Blog](https://blog.gradle.org/)
* [Newsletter](https://newsletter.gradle.com/)
* [Twitter](https://twitter.com/gradle)

Products

* [Build Scans](https://gradle.com/build-scans)
* [Build Cache](https://gradle.com/build-cache)
* [Enterprise Docs](https://gradle.com/enterprise/resources)

Get Help

* [Forums](https://discuss.gradle.org/c/help-discuss)
* [GitHub](https://github.com/gradle/)
* [Training](https://gradle.org/training/)
* [Services](https://gradle.org/services/)

Subscribe for important Gradle updates and news

Subscribe

By entering your email, you agree to our [Terms](https://gradle.org/terms/) and [Privacy Policy](https://gradle.org/privacy/), including receipt of emails. You can unsubscribe at any time.

© [Gradle Inc.](https://gradle.com) 2018 All rights reserved.

[Careers](https://gradle.com/careers) | [Privacy](https://gradle.org/privacy) | [Terms of Service](https://gradle.org/terms) | [Contact](https://gradle.org/contact/)