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/)

# Build Cache

Contents

[Overview](#4d34og8)

[Enable the Build Cache](#2s8eyo1)

[Task Output Caching](#17dp8vu)

[Cacheable tasks](#3rdcrjn)

[Configure the Build Cache](#26in1rg)

[How to set up an HTTP build cache backend](#lnxbz9)

[Implement your own Build Cache](#35nkun2)

| **✨** | The build cache feature described here is different from the [Android plugin build cache](https://developer.android.com/studio/build/build-cache.html). |
| --- | --- |

[Overview](#4d34og8)

The Gradle *build cache* is a cache mechanism that aims to save time by reusing outputs produced by other builds. The build cache works by storing (locally or remotely) build outputs and allowing builds to fetch these outputs from the cache when it is determined that inputs have not changed, avoiding the expensive work of regenerating them.

A first feature using the build cache is *task output caching*. Essentially, task output caching leverages the same intelligence as [up-to-date checks](http://docs.google.com/more_about_tasks.html#sec:up_to_date_checks) that Gradle uses to avoid work when a previous local build has already produced a set of task outputs. But instead of being limited to the previous build in the same workspace, task output caching allows Gradle to reuse task outputs from any earlier build in any location on the local machine. When using a shared build cache for task output caching this even works across developer machines and build agents.

Apart from task output caching, we expect other features to use the build cache in the future.

| **✨** | A complete guide is available about [using the build cache](https://guides.gradle.org/using-build-cache/). It covers the different scenarios caching can improve, and detailed discussions of the different caveats you need to be aware of when enabling caching for a build. |
| --- | --- |

[Enable the Build Cache](#2s8eyo1)

By default, the build cache is not enabled. You can enable the build cache in a couple of ways:

Run with --build-cache on the command-line

Gradle will use the build cache for this build only.

Put org.gradle.caching=true in your gradle.properties

Gradle will try to reuse outputs from previous builds for all builds, unless explicitly disabled with --no-build-cache.

When the build cache is enabled, it will store build outputs in the Gradle user home. For configuring this directory or different kinds of build caches see [Configure the Build Cache](#26in1rg).

[Task Output Caching](#17dp8vu)

Beyond incremental builds described in [up-to-date checks](http://docs.google.com/more_about_tasks.html#sec:up_to_date_checks), Gradle can save time by reusing outputs from previous executions of a task by matching inputs to the task. Task outputs can be reused between builds on one computer or even between builds running on different computers via a build cache.

We have focused on the use case where users have an organization-wide remote build cache that is populated regularly by continuous integration builds. Developers and other continuous integration agents should load cache entries from the remote build cache. We expect that developers will not be allowed to populate the remote build cache, and all continuous integration builds populate the build cache after running the clean task.

For your build to play well with task output caching it must work well with the [incremental build](http://docs.google.com/more_about_tasks.html#sec:up_to_date_checks) feature. For example, when running your build twice in a row all tasks with outputs should be UP-TO-DATE. You cannot expect faster builds or correct builds when enabling task output caching when this prerequisite is not met.

Task output caching is automatically enabled when you enable the build cache, see [Enable the Build Cache](#2s8eyo1).

[What does it look like](#1ksv4uv)

Let us start with a project using the Java plugin which has a few Java source files. We run the build the first time.

> gradle --build-cache compileJava  
:compileJava  
:processResources  
:classes  
:jar  
:assemble  
  
BUILD SUCCESSFUL

We see the directory used by the local build cache in the output. Apart from that the build was the same as without the build cache. Let’s clean and run the build again.

> gradle clean  
:clean  
  
BUILD SUCCESSFUL

> gradle --build-cache assemble  
:compileJava FROM-CACHE  
:processResources  
:classes  
:jar  
:assemble  
  
BUILD SUCCESSFUL

Now we see that, instead of executing the :compileJava task, the outputs of the task have been loaded from the build cache. The other tasks have not been loaded from the build cache since they are not cacheable. This is due to :classes and :assemble being [lifecycle tasks](http://docs.google.com/more_about_tasks.html#sec:lifecycle_tasks) and :processResources and :jar being Copy-like tasks which are not cacheable since it is generally faster to execute them.

[Cacheable tasks](#3rdcrjn)

Since a task describes all of its inputs and outputs, Gradle can compute a *build cache key* that uniquely defines the task’s outputs based on its inputs. That build cache key is used to request previous outputs from a build cache or store new outputs in the build cache. If the previous build outputs have been already stored in the cache by someone else, e.g. your continuous integration server or other developers, you can avoid executing most tasks locally.

The following inputs contribute to the build cache key for a task in the same way that they do for [up-to-date checks](http://docs.google.com/more_about_tasks.html#sec:how_does_it_work):

* The task type and its classpath
* The names of the output properties
* The names and values of properties annotated as described in [the section called "Custom task types"](http://docs.google.com/more_about_tasks.html#sec:task_input_output_annotations)
* The names and values of properties added by the DSL via [TaskInputs](http://docs.google.com/javadoc/org/gradle/api/tasks/TaskInputs.html)
* The classpath of the Gradle distribution, buildSrc and plugins
* The content of the build script when it affects execution of the task

Task types need to opt-in to task output caching using the [@CacheableTask](http://docs.google.com/javadoc/org/gradle/api/tasks/CacheableTask.html) annotation. Note that [@CacheableTask](http://docs.google.com/javadoc/org/gradle/api/tasks/CacheableTask.html) is not inherited by subclasses. Custom task types are *not* cacheable by default.

[Built-in cacheable tasks](#44sinio)

Currently, the following built-in Gradle tasks are cacheable:

* Java toolchain: [JavaCompile](http://docs.google.com/dsl/org.gradle.api.tasks.compile.JavaCompile.html), [Javadoc](http://docs.google.com/dsl/org.gradle.api.tasks.javadoc.Javadoc.html)
* Groovy toolchain: [GroovyCompile](http://docs.google.com/dsl/org.gradle.api.tasks.compile.GroovyCompile.html), [Groovydoc](http://docs.google.com/dsl/org.gradle.api.tasks.javadoc.Groovydoc.html)
* Scala toolchain: [ScalaCompile](http://docs.google.com/dsl/org.gradle.api.tasks.scala.ScalaCompile.html), [PlatformScalaCompile](http://docs.google.com/javadoc/org/gradle/language/scala/tasks/PlatformScalaCompile.html), [ScalaDoc](http://docs.google.com/dsl/org.gradle.api.tasks.scala.ScalaDoc.html)
* Native toolchain: [CppCompile](http://docs.google.com/javadoc/org/gradle/language/cpp/tasks/CppCompile.html), [CCompile](http://docs.google.com/javadoc/org/gradle/language/c/tasks/CCompile.html), [SwiftCompile](http://docs.google.com/javadoc/org/gradle/language/swift/tasks/SwiftCompile.html)
* Testing: [Test](http://docs.google.com/dsl/org.gradle.api.tasks.testing.Test.html)
* Code quality tasks: [Checkstyle](http://docs.google.com/dsl/org.gradle.api.plugins.quality.Checkstyle.html), [CodeNarc](http://docs.google.com/dsl/org.gradle.api.plugins.quality.CodeNarc.html), [FindBugs](http://docs.google.com/dsl/org.gradle.api.plugins.quality.FindBugs.html), [JDepend](http://docs.google.com/dsl/org.gradle.api.plugins.quality.JDepend.html), [Pmd](http://docs.google.com/dsl/org.gradle.api.plugins.quality.Pmd.html)
* JaCoCo: [JacocoMerge](http://docs.google.com/dsl/org.gradle.testing.jacoco.tasks.JacocoMerge.html), [JacocoReport](http://docs.google.com/dsl/org.gradle.testing.jacoco.tasks.JacocoReport.html)
* Other tasks: [AntlrTask](http://docs.google.com/dsl/org.gradle.api.plugins.antlr.AntlrTask.html), [ValidateTaskProperties](http://docs.google.com/javadoc/org/gradle/plugin/devel/tasks/ValidateTaskProperties.html), [WriteProperties](http://docs.google.com/dsl/org.gradle.api.tasks.WriteProperties.html)

All other built-in tasks are currently not cacheable.

Some tasks, like [Copy](http://docs.google.com/dsl/org.gradle.api.tasks.Copy.html) or [Jar](http://docs.google.com/dsl/org.gradle.api.tasks.bundling.Jar.html), usually do not make sense to make cacheable because Gradle is only copying files from one location to another. It also doesn’t make sense to make tasks cacheable that do not produce outputs or have no task actions.

[Third party plugins](#2jxsxqh)

There are third party plugins that work well with the build cache. The most prominent examples are the [Android plugin 3.1+](https://developer.android.com/studio/releases/gradle-plugin.html) and the [Kotlin plugin 1.2.21+](https://blog.gradle.org/kotlin-build-cache-use). For other third party plugins, check their documentation to find out whether they support the build cache.

[Declaring task inputs and outputs](#z337ya)

It is very important that a cacheable task has a complete picture of its inputs and outputs, so that the results from one build can be safely re-used somewhere else.

Missing task inputs can cause incorrect cache hits, where different results are treated as identical because the same cache key is used by both executions. Missing task outputs can cause build failures if Gradle does not completely capture all outputs for a given task. Wrongly declared task inputs can lead to cache misses especially when containing volatile data or absolute paths. (See [the section called "Task inputs and outputs"](http://docs.google.com/more_about_tasks.html#sec:task_inputs_outputs) on what should be declared as inputs and outputs.)

| **✨** | The task path is *not* an input to the build cache key. This means that tasks with different task paths can re-use each other’s outputs as long as Gradle determines that executing them yields the same result. |
| --- | --- |

In order to ensure that the inputs and outputs are properly declared use integration tests (for example using TestKit) to check that a task produces the same outputs for identical inputs and captures all output files for the task. We suggest adding tests to ensure that the task inputs are relocatable, i.e. that the task can be loaded from the cache into a different build directory (see [@PathSensitive](http://docs.google.com/javadoc/org/gradle/api/tasks/PathSensitive.html)).

In order to handle volatile inputs for your tasks consider [configuring input normalization](http://docs.google.com/more_about_tasks.html#sec:configure_input_normalization).

[Configure the Build Cache](#26in1rg)

You can configure the build cache by using the [Settings.buildCache(org.gradle.api.Action)](http://docs.google.com/dsl/org.gradle.api.initialization.Settings.html#org.gradle.api.initialization.Settings:buildCache(org.gradle.api.Action)) block in settings.gradle.

Gradle supports a local and a remote build cache that can be configured separately. When both build caches are enabled, Gradle tries to load build outputs from the local build cache first, and then tries the remote build cache if no build outputs are found. If outputs are found in the remote cache, they are also stored in the local cache, so next time they will be found locally. Gradle stores ("pushes") build outputs in any build cache that is enabled and has [BuildCache.isPush()](http://docs.google.com/javadoc//org/gradle/caching/configuration/BuildCache.html#isPush--) set to true.

By default, the local build cache has push enabled, and the remote build cache has push disabled.

The local build cache is pre-configured to be a [DirectoryBuildCache](http://docs.google.com/dsl/org.gradle.caching.local.DirectoryBuildCache.html) and enabled by default. The remote build cache can be configured by specifying the type of build cache to connect to ([BuildCacheConfiguration.remote(java.lang.Class)](http://docs.google.com/dsl/org.gradle.caching.configuration.BuildCacheConfiguration.html#org.gradle.caching.configuration.BuildCacheConfiguration:remote(java.lang.Class))).

[Built-in local build cache](#3j2qqm3)

The built-in local build cache, [DirectoryBuildCache](http://docs.google.com/dsl/org.gradle.caching.local.DirectoryBuildCache.html), uses a directory to store build cache artifacts. By default, this directory resides in the Gradle user home directory, but its location is configurable.

Gradle will periodically clean-up the local cache directory by removing entries that have not been used recently to conserve disk space.

For more details on the configuration options refer to the DSL documentation of [DirectoryBuildCache](http://docs.google.com/dsl/org.gradle.caching.local.DirectoryBuildCache.html). Here is an example of the configuration.

[Example: Configure the local cache](#1y810tw)

**settings.gradle**

buildCache {  
 local(DirectoryBuildCache) {  
 directory = new File(rootDir, 'build-cache')  
 removeUnusedEntriesAfterDays = 30  
 }  
}

[Remote HTTP build cache](#4i7ojhp)

Gradle has built-in support for connecting to a remote build cache backend via HTTP. For more details on what the protocol looks like see [HttpBuildCache](http://docs.google.com/dsl/org.gradle.caching.http.HttpBuildCache.html). Note that by using the following configuration the local build cache will be used for storing build outputs while the local and the remote build cache will be used for retrieving build outputs.

[Example: Load from HttpBuildCache](#2xcytpi)

**settings.gradle**

buildCache {  
 remote(HttpBuildCache) {  
 url = 'https://example.com:8123/cache/'  
 }  
}

You can configure the credentials the [HttpBuildCache](http://docs.google.com/dsl/org.gradle.caching.http.HttpBuildCache.html) uses to access the build cache server as shown in the following example.

[Example: Configure remote HTTP cache](#1ci93xb)

**settings.gradle**

buildCache {  
 remote(HttpBuildCache) {  
 url = 'http://example.com:8123/cache/'  
 credentials {  
 username = 'build-cache-user'  
 password = 'some-complicated-password'  
 }  
 }  
}

| **✨** | You may encounter problems with an untrusted SSL certificate when you try to use a build cache backend with an HTTPS URL. The ideal solution is for someone to add a valid SSL certificate to the build cache backend, but we recognize that you may not be able to do that. In that case, set [HttpBuildCache.isAllowUntrustedServer()](http://docs.google.com/dsl/org.gradle.caching.http.HttpBuildCache.html#org.gradle.caching.http.HttpBuildCache:allowUntrustedServer) to true:  **settings.gradle**  buildCache {  remote(HttpBuildCache) {  url = 'https://example.com:8123/cache/'  allowUntrustedServer = true  } }  This is a convenient workaround, but you shouldn’t use it as a long-term solution. |
| --- | --- |

[Configuration use cases](#3whwml4)

The recommended use case for the build cache is that your continuous integration server populates the remote build cache from clean builds while developers load from the remote build cache and store in the local build cache. The configuration would then look as follows.

[Example: Recommended setup for CI push use case](#2bn6wsx)

**settings.gradle**

ext.isCiServer = System.getenv().containsKey("CI")  
  
buildCache {  
 local {  
 enabled = !isCiServer  
 }  
 remote(HttpBuildCache) {  
 url = 'https://example.com:8123/cache/'  
 push = isCiServer  
 }  
}

If you use a buildSrc directory, you should make sure that it uses the same build cache configuration as the main build. This can be achieved by applying the same script to buildSrc/settings.gradle and settings.gradle as shown in the following example.

[Example: Consistent setup for buildSrc and main build](#qsh70q)

**settings.gradle**

apply from: new File(settingsDir, 'gradle/buildCacheSettings.gradle')

**gradle/buildCacheSettings.gradle**

ext.isCiServer = System.getenv().containsKey("CI")  
  
buildCache {  
 local {  
 enabled = !isCiServer  
 }  
 remote(HttpBuildCache) {  
 url = 'https://example.com:8123/cache/'  
 push = isCiServer  
 }  
}

**buildSrc/settings.gradle**

apply from: new File(settingsDir, '../gradle/buildCacheSettings.gradle')

It is also possible to configure the build cache from an [init script](http://docs.google.com/init_scripts.html#sec:using_an_init_script), which can be used from the command line, added to your Gradle user home or be a part of your custom Gradle distribution.

[Example: Init script to configure the build cache](#3as4poj)

**init.gradle**

gradle.settingsEvaluated { settings ->  
 settings.buildCache {  
 // vvv Your custom configuration goes here  
 remote(HttpBuildCache) {  
 url = 'https://example.com:8123/cache/'  
 }  
 // ^^^ Your custom configuration goes here  
 }  
}

[Build cache and composite builds](#1pxezwc)

Gradle’s [composite build feature](http://docs.google.com/composite_builds.html#composite_builds) allows including other complete Gradle builds into another. Such included builds will inherit the build cache configuration from the top level build, regardless of whether the included builds define build cache configuration themselves or not.

The build cache configuration present for any included build is effectively ignored, in favour of the top level build’s configuration. This also applies to any buildSrc projects of any included builds.

[How to set up an HTTP build cache backend](#lnxbz9)

Gradle provides a Docker image for a [build cache node](https://hub.docker.com/r/gradle/build-cache-node/), which can connect with Gradle Enterprise for centralized management. The cache node can also be used without a Gradle Enterprise installation with restricted functionality.

[Implement your own Build Cache](#35nkun2)

Using a different build cache backend to store build outputs (which is not covered by the built-in support for connecting to an HTTP backend) requires implementing your own logic for connecting to your custom build cache backend. To this end, custom build cache types can be registered via [BuildCacheConfiguration.registerBuildCacheService(java.lang.Class, java.lang.Class)](http://docs.google.com/javadoc/org/gradle/caching/configuration/BuildCacheConfiguration.html#registerBuildCacheService-java.lang.Class-java.lang.Class-).

[Gradle Enterprise](https://gradle.com/build-cache) includes a high-performance, easy to install and operate, shared build cache backend.

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/)