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/)

# Comparing Builds

Contents

[Definition of terms](#4d34og8)

[Current Capabilities](#2s8eyo1)

[Comparing Gradle Builds](#17dp8vu)

| **✨** | Build comparison support is an [incubating](http://docs.google.com/feature_lifecycle.html#feature_lifecycle) feature. This means that it is incomplete and not yet at regular Gradle production quality. This also means that this Gradle User Guide chapter is a work in progress. |
| --- | --- |

Gradle provides support for comparing the *outcomes* (e.g. the produced binary archives) of two builds. There are several reasons why you may want to compare the outcomes of two builds. You may want to compare:

* A build with a newer version of Gradle than it’s currently using (i.e. upgrading the Gradle version).
* A Gradle build with a build executed by another tool such as Apache Ant, Apache Maven or something else (i.e. migrating to Gradle).
* The same Gradle build, with the same version, before and after a change to the build (i.e. testing build changes).

By comparing builds in these scenarios you can make an informed decision about the Gradle upgrade, migration to Gradle or build change by understanding the differences in the outcomes. The comparison process produces a HTML report outlining which outcomes were found to be identical and identifying the differences between non-identical outcomes.

[Definition of terms](#4d34og8)

The following are the terms used for build comparison and their definitions.

“Build”

In the context of build comparison, a build is not necessarily a Gradle build. It can be any invokable “process” that produces observable “outcomes”. At least one of the builds in a comparison will be a Gradle build.

“Build Outcome”

Something that happens in an observable manner during a build, such as the creation of a zip file or test execution. These are the things that are compared.

“Source Build”

The build that comparisons are being made against, typically the build in its “current” state. In other words, the left hand side of the comparison.

“Target Build”

The build that is being compared to the source build, typically the “proposed” build. In other words, the right hand side of the comparison.

“Host Build”

The Gradle build that executes the comparison process. It may be the same project as either the “target” or “source” build or may be a completely separate project. It does not need to be the same Gradle version as the “source” or “target” builds. The host build must be run with Gradle 1.2 or newer.

“Compared Build Outcome”

Build outcomes that are intended to be logically equivalent in the “source” and “target” builds, and are therefore meaningfully comparable.

“Uncompared Build Outcome”

A build outcome is uncompared if a logical equivalent from the other build cannot be found (e.g. a build produces a zip file that the other build does not).

“Unknown Build Outcome”

A build outcome that cannot be understood by the host build. This can occur when the source or target build is a newer Gradle version than the host build and that Gradle version exposes new outcome types. Unknown build outcomes can be compared in so far as they can be identified to be logically equivalent to an unknown build outcome in the other build, but no meaningful comparison of what the build outcome actually is can be performed. Using the latest Gradle version for the host build will avoid encountering unknown build outcomes.

[Current Capabilities](#2s8eyo1)

As this is an [incubating](http://docs.google.com/feature_lifecycle.html#feature_lifecycle) feature, a limited set of the eventual functionality has been implemented at this time.

[Supported builds](#3rdcrjn)

Only support for comparing Gradle builds is available at this time. Both the source and target build must execute with Gradle newer or equal to version 1.0. The host build must be at least version 1.2. If the host build is run with version 3.0 or newer, source and target builds must be at least version 1.2. If the host build is run with a version older than 2.0, source and target builds must be older than version 3.0. So if you for example want to compare a build under version 1.1 with a build under version 3.0, you have to execute the host build with a 2.x version.

Future versions will provide support for executing builds from other build systems such as Apache Ant or Apache Maven, as well as support for executing arbitrary processes (e.g. shell script based builds)

[Supported build outcomes](#26in1rg)

Only support for comparing build outcomes that are zip archives is supported at this time. This includes jar, war and ear archives.

Future versions will provide support for comparing outcomes such as test execution (i.e. which tests were executed, which tests failed, etc.)

[Comparing Gradle Builds](#17dp8vu)

The compare-gradle-builds plugin can be used to facilitate a comparison between two Gradle builds. The plugin adds a [CompareGradleBuilds](http://docs.google.com/dsl/org.gradle.api.plugins.buildcomparison.gradle.CompareGradleBuilds.html) task named “compareGradleBuilds” to the project. The configuration of this task specifies what is to be compared. By default, it is configured to compare the current build with itself using the current Gradle version by executing the tasks: “clean assemble”.

apply plugin: 'compare-gradle-builds'

This task can be configured to change what is compared.

compareGradleBuilds {  
 sourceBuild {  
 projectDir "/projects/project-a"  
 gradleVersion "1.1"  
 }  
 targetBuild {  
 projectDir "/projects/project-b"  
 gradleVersion "1.2"  
 }  
}

The example above specifies a comparison between two different projects using two different Gradle versions.

[Trying Gradle upgrades](#lnxbz9)

You can use the build comparison functionality to very quickly try a new Gradle version with your build.

To try your current build with a different Gradle version, simply add the following to the build.gradle of the *root project*.

apply plugin: 'compare-gradle-builds'  
  
compareGradleBuilds {  
 targetBuild.gradleVersion = "«gradle version»"  
}

Then simply execute the compareGradleBuilds task. You will see the console output of the “source” and “target” builds as they are executing.

[The comparison “result”](#35nkun2)

If there are any differences between the *compared outcomes*, the task will fail. The location of the HTML report providing insight into the comparison will be given. If all compared outcomes are found to be identical, and there are no uncompared outcomes, and there are no unknown build outcomes, the task will succeed.

You can configure the task to not fail on compared outcome differences by setting the ignoreFailures property to true.

compareGradleBuilds {  
 ignoreFailures = true  
}

[Which archives are compared?](#1ksv4uv)

For an archive to be a candidate for comparison, it must be added as an artifact of the archives configuration. Take a look at [Legacy Publishing](http://docs.google.com/artifact_management.html#artifact_management) for more information on how to configure and add artifacts.

The archive must also have been produced by a [Zip](http://docs.google.com/dsl/org.gradle.api.tasks.bundling.Zip.html), [Jar](http://docs.google.com/dsl/org.gradle.api.tasks.bundling.Jar.html), [War](http://docs.google.com/dsl/org.gradle.api.tasks.bundling.War.html), [Ear](http://docs.google.com/dsl/org.gradle.plugins.ear.Ear.html) task. Future versions of Gradle will support increased flexibility in this area.

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/)