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/)

# Logging

Contents

[Log levels](#4d34og8)

[Choosing a log level](#2s8eyo1)

[Writing your own log messages](#17dp8vu)

[Logging from external tools and libraries](#3rdcrjn)

[Changing what Gradle logs](#26in1rg)

The log is the main 'UI' of a build tool. If it is too verbose, real warnings and problems are easily hidden by this. On the other hand you need relevant information for figuring out if things have gone wrong. Gradle defines 6 log levels, as shown in [Log levels](#4d34og8). There are two Gradle-specific log levels, in addition to the ones you might normally see. Those levels are *QUIET* and *LIFECYCLE*. The latter is the default, and is used to report build progress.

[Log levels](#4d34og8)

| **ERROR** | Error messages |
| --- | --- |
| **QUIET** | Important information messages |
| **WARNING** | Warning messages |
| **LIFECYCLE** | Progress information messages |
| **INFO** | Information messages |
| **DEBUG** | Debug messages |

| **✨** | The rich components of the console (build status and work in progress area) are displayed regardless of the log level used. Before Gradle 4.0 those rich components were only displayed at log level LIFECYCLE or below. |
| --- | --- |

[Choosing a log level](#2s8eyo1)

You can use the command line switches shown in [Log level command-line options](#lnxbz9) to choose different log levels. You can also configure the log level using gradle.properties, see [Gradle properties](http://docs.google.com/build_environment.html#sec:gradle_configuration_properties). In [Stacktrace command-line options](#35nkun2) you find the command line switches which affect stacktrace logging.

Table 1. Log level command-line options

| **Option** | **Outputs Log Levels** |
| --- | --- |
| no logging options | LIFECYCLE and higher |
| -q or --quiet | QUIET and higher |
| -w or --warn | WARN and higher |
| -i or --info | INFO and higher |
| -d or --debug | DEBUG and higher (that is, all log messages) |

[Stacktrace command-line options](#35nkun2)

-s or --stacktrace

Truncated stacktraces are printed. We recommend this over full stacktraces. Groovy full stacktraces are extremely verbose (Due to the underlying dynamic invocation mechanisms. Yet they usually do not contain relevant information for what has gone wrong in *your* code.) This option renders stacktraces for deprecation warnings.

-S or --full-stacktrace

The full stacktraces are printed out. This option renders stacktraces for deprecation warnings.

<No stacktrace options>

No stacktraces are printed to the console in case of a build error (e.g. a compile error). Only in case of internal exceptions will stacktraces be printed. If the DEBUG log level is chosen, truncated stacktraces are always printed.

[Writing your own log messages](#17dp8vu)

A simple option for logging in your build file is to write messages to standard output. Gradle redirects anything written to standard output to its logging system at the QUIET log level.

[Example: Using stdout to write log messages](#1ksv4uv)

**build.gradle**

println 'A message which is logged at QUIET level'

Gradle also provides a logger property to a build script, which is an instance of [Logger](http://docs.google.com/javadoc/org/gradle/api/logging/Logger.html). This interface extends the SLF4J Logger interface and adds a few Gradle specific methods to it. Below is an example of how this is used in the build script:

[Example: Writing your own log messages](#44sinio)

**build.gradle**

logger.quiet('An info log message which is always logged.')  
logger.error('An error log message.')  
logger.warn('A warning log message.')  
logger.lifecycle('A lifecycle info log message.')  
logger.info('An info log message.')  
logger.debug('A debug log message.')  
logger.trace('A trace log message.')

Use the [typical SLF4J pattern](https://www.slf4j.org/manual.html#typical_usage) to replace a placeholder with an actual value as part of the log message.

[Example: Writing a log message with placeholder](#2jxsxqh)

**build.gradle**

logger.info('A {} log message', 'info')

You can also hook into Gradle’s logging system from within other classes used in the build (classes from the buildSrc directory for example). Simply use an SLF4J logger. You can use this logger the same way as you use the provided logger in the build script.

[Example: Using SLF4J to write log messages](#z337ya)

**build.gradle**

import org.slf4j.Logger  
import org.slf4j.LoggerFactory  
  
Logger slf4jLogger = LoggerFactory.getLogger('some-logger')  
slf4jLogger.info('An info log message logged using SLF4j')

[Logging from external tools and libraries](#3rdcrjn)

Internally, Gradle uses Ant and Ivy. Both have their own logging system. Gradle redirects their logging output into the Gradle logging system. There is a 1:1 mapping from the Ant/Ivy log levels to the Gradle log levels, except the Ant/Ivy TRACE log level, which is mapped to Gradle DEBUG log level. This means the default Gradle log level will not show any Ant/Ivy output unless it is an error or a warning.

There are many tools out there which still use standard output for logging. By default, Gradle redirects standard output to the QUIET log level and standard error to the ERROR level. This behavior is configurable. The project object provides a [LoggingManager](http://docs.google.com/javadoc/org/gradle/api/logging/LoggingManager.html), which allows you to change the log levels that standard out or error are redirected to when your build script is evaluated.

[Example: Configuring standard output capture](#3j2qqm3)

**build.gradle**

logging.captureStandardOutput LogLevel.INFO  
println 'A message which is logged at INFO level'

To change the log level for standard out or error during task execution, tasks also provide a [LoggingManager](http://docs.google.com/javadoc/org/gradle/api/logging/LoggingManager.html).

[Example: Configuring standard output capture for a task](#1y810tw)

**build.gradle**

task logInfo {  
 logging.captureStandardOutput LogLevel.INFO  
 doFirst {  
 println 'A task message which is logged at INFO level'  
 }  
}

Gradle also provides integration with the Java Util Logging, Jakarta Commons Logging and Log4j logging toolkits. Any log messages which your build classes write using these logging toolkits will be redirected to Gradle’s logging system.

[Changing what Gradle logs](#26in1rg)

You can replace much of Gradle’s logging UI with your own. You might do this, for example, if you want to customize the UI in some way - to log more or less information, or to change the formatting. You replace the logging using the [Gradle.useLogger(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.invocation.Gradle.html#org.gradle.api.invocation.Gradle:useLogger(java.lang.Object)) method. This is accessible from a build script, or an init script, or via the embedding API. Note that this completely disables Gradle’s default output. Below is an example init script which changes how task execution and build completion is logged.

[Example: Customizing what Gradle logs](#4i7ojhp)

**init.gradle**

useLogger(new CustomEventLogger())  
  
class CustomEventLogger extends BuildAdapter implements TaskExecutionListener {  
  
 public void beforeExecute(Task task) {  
 println "[$task.name]"  
 }  
  
 public void afterExecute(Task task, TaskState state) {  
 println()  
 }  
  
 public void buildFinished(BuildResult result) {  
 println 'build completed'  
 if (result.failure != null) {  
 result.failure.printStackTrace()  
 }  
 }  
}

**Output of** gradle -I init.gradle build

> gradle -I init.gradle build  
  
> Task :compile  
[compile]  
compiling source  
  
  
> Task :testCompile  
[testCompile]  
compiling test source  
  
  
> Task :test  
[test]  
running unit tests  
  
  
> Task :build  
[build]  
  
build completed  
3 actionable tasks: 3 executed

Your logger can implement any of the listener interfaces listed below. When you register a logger, only the logging for the interfaces that it implements is replaced. Logging for the other interfaces is left untouched. You can find out more about the listener interfaces in [Build lifecycle events](http://docs.google.com/build_lifecycle.html#build_lifecycle_events).

* [BuildListener](http://docs.google.com/javadoc/org/gradle/BuildListener.html)
* [ProjectEvaluationListener](http://docs.google.com/javadoc/org/gradle/api/ProjectEvaluationListener.html)
* [TaskExecutionGraphListener](http://docs.google.com/javadoc/org/gradle/api/execution/TaskExecutionGraphListener.html)
* [TaskExecutionListener](http://docs.google.com/javadoc/org/gradle/api/execution/TaskExecutionListener.html)
* [TaskActionListener](http://docs.google.com/javadoc/org/gradle/api/execution/TaskActionListener.html)

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/)