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/)

# The Gradle Wrapper

Contents

[Adding the Gradle Wrapper](#4d34og8)

[Using the Gradle Wrapper](#2s8eyo1)

[Upgrading the Gradle Wrapper](#17dp8vu)

[Customizing the Gradle Wrapper](#3rdcrjn)

The recommended way to execute any Gradle build is with the help of the Gradle Wrapper (in short just “Wrapper”). The Wrapper is a script that invokes a declared version of Gradle, downloading it beforehand if necessary. As a result, developers can get up and running with a Gradle project quickly without having to follow manual installation processes saving your company time and money.



*Figure 1. The Wrapper workflow*

In a nutshell you gain the following benefits:

* Standardizes a project on a given Gradle version, leading to more reliable and robust builds.
* Provisioning a new Gradle version to different users and execution environment (e.g. IDEs or Continuous Integration servers) is as simple as changing the Wrapper definition.

So how does it work? For a user there are typically three different workflows:

* You set up a new Gradle project and want to [add the Wrapper](#4d34og8) to it.
* You want to [run a project with the Wrapper](#2s8eyo1) that already provides it.
* You want to [upgrade the Wrapper](#17dp8vu) to a new version of Gradle.

The following sections explain each of these use cases in more detail.

[Adding the Gradle Wrapper](#4d34og8)

Generating the Wrapper files requires an installed version of the Gradle runtime on your machine as described in [Installation](http://docs.google.com/installation.html#installation). Thankfully, generating the initial Wrapper files is a one-time process.

Every vanilla Gradle build comes with a built-in task called wrapper. You’ll be able to find the task listed under the group "Build Setup tasks" when [listing the tasks](http://docs.google.com/command_line_interface.html#sec:listing_tasks). Executing the wrapper task generates the necessary Wrapper files in the project directory.

[Example: Running the Wrapper task](#26in1rg)

**Output of** gradle wrapper

> gradle wrapper  
> Task :wrapper  
  
BUILD SUCCESSFUL in 0s  
1 actionable task: 1 executed

| **✨** | To make the Wrapper files available to other developers and execution environments you’ll need to check them into version control. All Wrapper files including the JAR file are very small in size. Adding the JAR file to version control is expected. Some organizations do not allow projects to submit binary files to version control. At the moment there are no alternative options to the approach. |
| --- | --- |

The generated Wrapper properties file, gradle/wrapper/gradle-wrapper.properties, stores the information about the Gradle distribution.

* The server hosting the Gradle distribution.
* The type of Gradle distribution. By default that’s the -bin distribution containing only the runtime but no sample code and documentation.
* The Gradle version used for executing the build. By default the wrapper task picks the exact same Gradle version that was used to generate the Wrapper files.

**gradle/wrapper/gradle-wrapper.properties**

distributionUrl=https\://services.gradle.org/distributions/gradle-4.3.1-bin.zip

All of those aspects are configurable at the time of generating the Wrapper files with the help of the following command line options.

--gradle-version

The Gradle version used for downloading and executing the Wrapper.

--distribution-type

The Gradle distribution type used for the Wrapper. Available options are bin and all. The default value is bin.

--gradle-distribution-url

The full URL pointing to Gradle distribution ZIP file. Using this option makes --gradle-version and --distribution-type obsolete as the URL already contains this information. This option is extremely valuable if you want to host the Gradle distribution inside your company’s network.

--gradle-distribution-sha256-sum

The SHA256 hash sum used for [verifying the downloaded Gradle distribution](#lnxbz9).

Let’s assume the following use case to illustrate the use of the command line options. You would like to generate the Wrapper with version 4.0 and use the -all distribution to enable your IDE to enable code-completion and being able to navigate to the Gradle source code. Those requirements are captured by the following command line execution:

[Example: Providing options to Wrapper task](#35nkun2)

**Output of** gradle wrapper --gradle-version 4.0 --distribution-type all

> gradle wrapper --gradle-version 4.0 --distribution-type all  
> Task :wrapper  
  
BUILD SUCCESSFUL in 0s  
1 actionable task: 1 executed

As a result you can find the desired information in the Wrapper properties file.

**gradle/wrapper/gradle-wrapper.properties**

distributionUrl=https\://services.gradle.org/distributions/gradle-4.0-all.zip

Let’s have a look at the following project layout to illustrate the expected Wrapper files:

.  
├── build.gradle  
├── settings.gradle  
├── gradle  
│ └── wrapper  
│ ├── gradle-wrapper.jar  
│ └── gradle-wrapper.properties  
├── gradlew  
└── gradlew.bat

A Gradle project typically provides a build.gradle and a settings.gradle file. The Wrapper files live alongside in the gradle directory and the root directory of the project. The following list explains their purpose.

gradle-wrapper.jar

The Wrapper JAR file containing code for downloading the Gradle distribution.

gradle-wrapper.properties

A properties file responsible for configuring the Wrapper runtime behavior e.g. the Gradle version compatible with this version.

gradlew, gradlew.bat

A shell script and a Windows batch script for executing the build with the Wrapper.

You can go ahead and [execute the build with the Wrapper](#2s8eyo1) without having to install the Gradle runtime. If the project you are working on does not contain those Wrapper files then you’ll need to [generate them](#4d34og8).

[Using the Gradle Wrapper](#2s8eyo1)

It is recommended to always execute a build with the Wrapper to ensure a reliable, controlled and standardized execution of the build. Using the Wrapper looks almost exactly like running the build with a Gradle installation. Depending on the operating system you either run gradlew or gradlew.bat instead of the gradle command. The following console output demonstrate the use of the Wrapper on a Windows machine for a Java-based project.

[Example: Executing the build with the Wrapper batch file](#1ksv4uv)

**Output of gradlew.bat build**

> gradlew.bat build  
Downloading https://services.gradle.org/distributions/gradle-4.0-all.zip  
.....................................................................................  
Unzipping C:\Documents and Settings\Claudia\.gradle\wrapper\dists\gradle-4.0-all\ac27o8rbd0ic8ih41or9l32mv\gradle-4.0-all.zip to C:\Documents and Settings\Claudia\.gradle\wrapper\dists\gradle-4.0-al\ac27o8rbd0ic8ih41or9l32mv  
Set executable permissions for: C:\Documents and Settings\Claudia\.gradle\wrapper\dists\gradle-4.0-all\ac27o8rbd0ic8ih41or9l32mv\gradle-4.0\bin\gradle  
  
BUILD SUCCESSFUL in 12s  
1 actionable task: 1 executed

In case the Gradle distribution is not available on the machine, the Wrapper will download it and store in the local file system. Any subsequent build invocation is going to reuse the existing local distribution as long as the distribution URL in the Gradle properties doesn’t change.

| **✨** | The Wrapper shell script and batch file reside in the root directory of a single or multi-project Gradle build. You will need to reference the correct path to those files in case you want to execute the build from a subproject directory e.g. ../../gradlew tasks. |
| --- | --- |

[Upgrading the Gradle Wrapper](#17dp8vu)

Projects will typically want to keep up with the times and upgrade their Gradle version to benefit from new features and improvements. One way to upgrade the Gradle version is manually change the distributionUrl property in the Wrapper property file. The better and recommended option is to run the wrapper task and provide the target Gradle version as described in [Adding the Gradle Wrapper](#4d34og8). Using the wrapper task ensures that any optimizations made to the Wrapper shell script or batch file with that specific Gradle version are applied to the project. As usual you’d commit the changes to the Wrapper files to version control.

Use the Gradle wrapper task to generate the wrapper, specifying a version. The default is the current version, which you can check by executing ./gradlew --version.

[Example: Upgrading the Wrapper version](#44sinio)

**Output of ./gradlew wrapper --gradle-version 4.2.1**

> ./gradlew wrapper --gradle-version 4.2.1  
  
BUILD SUCCESSFUL in 4s  
1 actionable task: 1 executed

[Example: Checking the Wrapper version after upgrading](#2jxsxqh)

**Output of ./gradlew -v**

> ./gradlew -v  
Downloading https://services.gradle.org/distributions/gradle-4.2.1-bin.zip  
...................................................................  
Unzipping /Users/claudia/.gradle/wrapper/dists/gradle-4.2.1-bin/dajvke9o8kmaxbu0kc5gcgeju/gradle-4.2.1-bin.zip to /Users/claudia/.gradle/wrapper/dists/gradle-4.2.1-bin/dajvke9o8kmaxbu0kc5gcgeju  
Set executable permissions for: /Users/claudia/.gradle/wrapper/dists/gradle-4.2.1-bin/dajvke9o8kmaxbu0kc5gcgeju/gradle-4.2.1/bin/gradle  
  
------------------------------------------------------------  
Gradle 4.2.1  
------------------------------------------------------------  
  
Build time: 2017-10-02 15:36:21 UTC  
Revision: a88ebd6be7840c2e59ae4782eb0f27fbe3405ddf  
  
Groovy: 2.4.12  
Ant: Apache Ant(TM) version 1.9.6 compiled on June 29 2015  
JVM: 1.8.0\_60 (Oracle Corporation 25.60-b23)  
OS: Mac OS X 10.13.1 x86\_64

[Customizing the Gradle Wrapper](#3rdcrjn)

Most users of Gradle are happy with the default runtime behavior of the Wrapper. However, organizational policies, security constraints or personal preferences might require you to dive deeper into customizing the Wrapper. Thankfully, the built-in wrapper task exposes numerous options to bend the runtime behavior to your needs. Most configuration options are exposed by the underlying task type [Wrapper](http://docs.google.com/dsl/org.gradle.api.tasks.wrapper.Wrapper.html).

Let’s assume you grew tired of defining the -all distribution type on the command line every time you upgrade the Wrapper. You can save yourself some keyboard strokes by re-configuring the wrapper task.

[Example: Customizing the Wrapper task](#z337ya)

**build.gradle**

wrapper {  
 distributionType = Wrapper.DistributionType.ALL  
}

With the configuration in place running ./gradlew wrapper --gradle-version 4.1 is enough to produce a distributionUrl value in the Wrapper properties file that will request the -all distribution.

**gradle/wrapper/gradle-wrapper.properties**

distributionUrl=https\://services.gradle.org/distributions/gradle-4.1-all.zip

Check out the API documentation for more detail descriptions of the available configuration options. You can also find various samples for configuring the Wrapper in the Gradle distribution.

[Authenticated Gradle distribution download](#3j2qqm3)

The Gradle Wrapper can download Gradle distributions from servers using HTTP Basic Authentication. This enables you to host the Gradle distribution on a private protected server. You can specify a username and password in two different ways depending on your use case: as system properties or directly embedded in the distributionUrl. Credentials in system properties take precedence over the ones embedded in distributionUrl.

| **💡** | Security Warning  HTTP Basic Authentication should only be used with HTTPS URLs and not plain HTTP ones. With Basic Authentication, the user credentials are sent in clear text. |
| --- | --- |

Using system properties can be done in the .gradle/gradle.properties file in the user’s home directory, or by other means, see [Gradle Configuration Properties](http://docs.google.com/build_environment.html#sec:gradle_configuration_properties).

**gradle.properties**

systemProp.gradle.wrapperUser=username  
systemProp.gradle.wrapperPassword=password

Embedding credentials in the distributionUrl in the gradle/wrapper/gradle-wrapper.properties file also works. Please note that this file is to be committed into your source control system. Shared credentials embedded in distributionUrl should only be used in a controlled environment.

**gradle/wrapper/gradle-wrapper.properties**

distributionUrl=https://username:password@somehost/path/to/gradle-distribution.zip

This can be used in conjunction with a proxy, authenticated or not. See [Accessing the web via a proxy](http://docs.google.com/build_environment.html#sec:accessing_the_web_via_a_proxy) for more information on how to configure the Wrapper to use a proxy.

[Verification of downloaded Gradle distributions](#lnxbz9)

The Gradle Wrapper allows for verification of the downloaded Gradle distribution via SHA-256 hash sum comparison. This increases security against targeted attacks by preventing a man-in-the-middle attacker from tampering with the downloaded Gradle distribution.

To enable this feature, download the .sha256 file associated with the Gradle distribution you want to verify.

[Downloading the SHA-256 file](#1y810tw)

You can download the .sha256 file from the [stable releases](https://services.gradle.org/distributions/) or [release candidate and nightly releases](https://services.gradle.org/distributions-snapshots/). The format of the file is a single line of text that is the SHA-256 hash of the corresponding zip file.

[Configuring checksum verification](#4i7ojhp)

Add the downloaded hash sum to gradle-wrapper.properties using the distributionSha256Sum property or use --gradle-distribution-sha256-sum on the command-line.

**gradle/wrapper/gradle-wrapper.properties**

distributionSha256Sum=371cb9fbebbe9880d147f59bab36d61eee122854ef8c9ee1ecf12b82368bcf10

Gradle will report a build failure in case the configured checksum does not match the checksum found on the server for hosting the distribution. Checksum Verification is only performed if the configured Wrapper distribution hasn’t been downloaded yet.

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/)