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 IDEA Plugin

Contents

[Usage](#4d34og8)

[Tasks](#2s8eyo1)

[Configuration](#17dp8vu)

[Customizing the generated files](#3rdcrjn)

[Further things to consider](#26in1rg)

The IDEA plugin generates files that are used by [IntelliJ IDEA](http://www.jetbrains.com/idea/), thus making it possible to open the project from IDEA (File - Open Project). Both external dependencies (including associated source and Javadoc files) and project dependencies are considered.

| **✨** | If you simply want to load a Gradle project into IntelliJ IDEA, then use the IDE’s [import facility](https://www.jetbrains.com/help/idea/gradle.html#gradle_import). You do not need to apply this plugin to import your project into IDEA, although if you do, the import will take account of any extra IDEA configuration you have that doesn’t directly modify the generated files — see the [Configuration](#17dp8vu) section for more details. |
| --- | --- |

What exactly the IDEA plugin generates depends on which other plugins are used:

Always

Generates an IDEA module file. Also generates an IDEA project and workspace file if the project is the root project.

[Java Plugin](http://docs.google.com/java_plugin.html#java_plugin)

Additionally adds Java configuration to the IDEA module and project files.

One focus of the IDEA plugin is to be open to customization. The plugin provides a standardized set of hooks for adding and removing content from the generated files.

[Usage](#4d34og8)

To use the IDEA plugin, include this in your build script:

[Example: Using the IDEA plugin](#lnxbz9)

**build.gradle**

apply plugin: 'idea'

The IDEA plugin adds a number of tasks to your project. The idea task generates an IDEA module file for the project. When the project is the root project, the idea task also generates an IDEA project and workspace. The IDEA project includes modules for each of the projects in the Gradle build.

The IDEA plugin also adds an openIdea task when the project is the root project. This task generates the IDEA configuration files and opens the result in IDEA. This means you can simply run ./gradlew openIdea from the root project to generate and open the IDEA project in one convenient step.

The IDEA plugin also adds a cleanIdea task to the project. This task deletes the generated files, if present.

[Tasks](#2s8eyo1)

The IDEA plugin adds the tasks shown below to a project. Notice that the clean task does not depend on the cleanIdeaWorkspace task. This is because the workspace typically contains a lot of user specific temporary data and it is not desirable to manipulate it outside IDEA.

idea

*Depends on*: ideaProject, ideaModule, ideaWorkspace

Generates all IDEA configuration files

openIdea

*Depends on*: idea

Generates all IDEA configuration files and opens the project in IDEA

cleanIdea — [Delete](http://docs.google.com/dsl/org.gradle.api.tasks.Delete.html)

*Depends on*: cleanIdeaProject, cleanIdeaModule

Removes all IDEA configuration files

cleanIdeaProject — [Delete](http://docs.google.com/dsl/org.gradle.api.tasks.Delete.html)

Removes the IDEA project file

cleanIdeaModule — [Delete](http://docs.google.com/dsl/org.gradle.api.tasks.Delete.html)

Removes the IDEA module file

cleanIdeaWorkspace — [Delete](http://docs.google.com/dsl/org.gradle.api.tasks.Delete.html)

Removes the IDEA workspace file

ideaProject — [GenerateIdeaProject](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.GenerateIdeaProject.html)

Generates the .ipr file. This task is only added to the root project.

ideaModule — [GenerateIdeaModule](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.GenerateIdeaModule.html)

Generates the .iml file

ideaWorkspace — [GenerateIdeaWorkspace](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.GenerateIdeaWorkspace.html)

Generates the .iws file. This task is only added to the root project.

[Configuration](#17dp8vu)

The plugin adds some configuration options that allow to customize the IDEA project and module files that it generates. These take the form of both model properties and lower-level mechanisms that modify the generated files directly. For example, you can add source and resource directories, as well as inject your own fragments of XML. The former type of configuration is honored by IDEA’s import facility, whereas the latter is not.

Here are the configuration properties you can use:

idea — [IdeaModel](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.model.IdeaModel.html)

Top level element that enables configuration of the idea plugin in a DSL-friendly fashion

idea.project [IdeaProject](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.model.IdeaProject.html)

Allows configuring project information

idea.module [IdeaModule](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.model.IdeaModule.html)

Allows configuring module information

idea.workspace [IdeaWorkspace](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.model.IdeaWorkspace.html)

Allows configuring the workspace XML

Follow the links to the types for examples of using these configuration properties.

[Customizing the generated files](#3rdcrjn)

The IDEA plugin provides hooks and behavior for customizing the generated content in a more controlled and detailed way. In addition, the withXml hook is the only practical way to modify the workspace file because its corresponding domain object is essentially empty.

| **✨** | The techniques we discuss in this section don’t work with IDEA’s import facility |
| --- | --- |

The tasks recognize existing IDEA files and merge them with the generated content.

[Merging](#35nkun2)

Sections of existing IDEA files that are also the target of generated content will be amended or overwritten, depending on the particular section. The remaining sections will be left as-is.

[Disabling merging with a complete overwrite](#1ksv4uv)

To completely rewrite existing IDEA files, execute a clean task together with its corresponding generation task, like “gradle cleanIdea idea” (in that order). If you want to make this the default behavior, add “tasks.idea.dependsOn(cleanIdea)” to your build script. This makes it unnecessary to execute the clean task explicitly.

This strategy can also be used for individual files that the plugin would generate. For instance, this can be done for the “.iml” file with “gradle cleanIdeaModule ideaModule”.

[Hooking into the generation lifecycle](#44sinio)

The plugin provides objects modeling the sections of the metadata files that are generated by Gradle. The generation lifecycle is as follows:

1. The file is read; or a default version provided by Gradle is used if it does not exist
2. The beforeMerged hook is executed with a domain object representing the existing file
3. The existing content is merged with the configuration inferred from the Gradle build or defined explicitly in the eclipse DSL
4. The whenMerged hook is executed with a domain object representing contents of the file to be persisted
5. The withXml hook is executed with a raw representation of the XML that will be persisted
6. The final XML is persisted

The following are the domain objects used for each of the model types:

[IdeaProject](http://docs.google.com/dsl/org.gradle.plugins.ide.idea.model.IdeaProject.html)

* beforeMerged { [Project](http://docs.google.com/javadoc/org/gradle/plugins/ide/idea/model/Project.html) arg -> …​ }
* whenMerged { [Project](http://docs.google.com/javadoc/org/gradle/plugins/ide/idea/model/Project.html) arg -> …​ }
* withXml { [XmlProvider](http://docs.google.com/javadoc/org/gradle/api/XmlProvider.html) arg -> …​ }

[IdeaModule](http://docs.google.com/javadoc/org/gradle/plugins/ide/idea/model/IdeaModule.html)

* beforeMerged { [Module](http://docs.google.com/javadoc/org/gradle/plugins/ide/idea/model/Module.html) arg -> …​ }
* whenMerged { [Module](http://docs.google.com/javadoc/org/gradle/plugins/ide/idea/model/Module.html) arg -> …​ }
* withXml { [XmlProvider](http://docs.google.com/javadoc/org/gradle/api/XmlProvider.html) arg -> …​ }

[IdeaWorkspace](http://docs.google.com/javadoc/org/gradle/plugins/ide/idea/model/IdeaWorkspace.html)

* beforeMerged { [Workspace](http://docs.google.com/javadoc/org/gradle/plugins/ide/idea/model/Workspace.html) arg -> …​ }
* whenMerged { [Workspace](http://docs.google.com/javadoc/org/gradle/plugins/ide/idea/model/Workspace.html) arg -> …​ }
* withXml { [XmlProvider](http://docs.google.com/javadoc/org/gradle/api/XmlProvider.html) arg -> …​ }

[Partial rewrite of existing content](#2jxsxqh)

A "complete rewrite" causes all existing content to be discarded, thereby losing any changes made directly in the IDE. The beforeMerged hook makes it possible to overwrite just certain parts of the existing content. The following example removes all existing dependencies from the Module domain object:

[Example: Partial Rewrite for Module](#z337ya)

**build.gradle**

idea.module.iml {  
 beforeMerged { module ->  
 module.dependencies.clear()  
 }  
}

The resulting module file will only contain Gradle-generated dependency entries, but not any other dependency entries that may have been present in the original file. (In the case of dependency entries, this is also the default behavior.) Other sections of the module file will be either left as-is or merged. The same could be done for the module paths in the project file:

[Example: Partial Rewrite for Project](#3j2qqm3)

**build.gradle**

idea.project.ipr {  
 beforeMerged { project ->  
 project.modulePaths.clear()  
 }  
}

[Modifying the fully populated domain objects](#1y810tw)

The whenMerged hook allows you to manipulate the fully populated domain objects. Often this is the preferred way to customize IDEA files. Here is how you would export all the dependencies of an IDEA module:

[Example: Export Dependencies](#4i7ojhp)

**build.gradle**

idea.module.iml {  
 whenMerged { module ->  
 module.dependencies\*.exported = true  
 }  
}

[Modifying the XML representation](#2xcytpi)

The withXml hook allows you to manipulate the in-memory XML representation just before the file gets written to disk. Although Groovy’s XML support makes up for a lot, this approach is less convenient than manipulating the domain objects. In return, you get total control over the generated file, including sections not modeled by the domain objects.

[Example: Customizing the XML](#1ci93xb)

**build.gradle**

idea.project.ipr {  
 withXml { provider ->  
 provider.node.component  
 .find { it.@name == 'VcsDirectoryMappings' }  
 .mapping.@vcs = 'Git'  
 }  
}

[Further things to consider](#26in1rg)

The paths of dependencies in the generated IDEA files are absolute. If you manually define a path variable pointing to the Gradle dependency cache, IDEA will automatically replace the absolute dependency paths with this path variable. you can configure this path variable via the “idea.pathVariables” property, so that it can do a proper merge without creating duplicates.

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/)