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/)

# Ivy Publish Plugin

Contents

[Usage](#4d34og8)

[Tasks](#2s8eyo1)

[Publications](#17dp8vu)

[Repositories](#3rdcrjn)

[Complete example](#26in1rg)

The Ivy Publish Plugin provides the ability to publish build artifacts in the [Apache Ivy](http://ant.apache.org/ivy/) format, usually to a repository for consumption by other builds or projects. What is published is one or more artifacts created by the build, and an Ivy *module descriptor* (normally ivy.xml) that describes the artifacts and the dependencies of the artifacts, if any.

A published Ivy module can be consumed by Gradle (see [Declaring Dependencies](http://docs.google.com/declaring_dependencies.html#declaring_dependencies)) and other tools that understand the Ivy format. You can learn about the fundamentals of publishing in [Publishing Overview](http://docs.google.com/publishing_overview.html#publishing_overview).

[Usage](#4d34og8)

To use the Ivy Publish Plugin, include the following in your build script:

[Example: Applying the Ivy Publish Plugin](#lnxbz9)

**build.gradle**

plugins {  
 id 'ivy-publish'  
}

The Ivy Publish Plugin uses an extension on the project named publishing of type [PublishingExtension](http://docs.google.com/dsl/org.gradle.api.publish.PublishingExtension.html). This extension provides a container of named publications and a container of named repositories. The Ivy Publish Plugin works with [IvyPublication](http://docs.google.com/dsl/org.gradle.api.publish.ivy.IvyPublication.html) publications and [IvyArtifactRepository](http://docs.google.com/dsl/org.gradle.api.artifacts.repositories.IvyArtifactRepository.html) repositories.

[Tasks](#2s8eyo1)

generateDescriptorFileFor*PubName*Publication — [GenerateIvyDescriptor](http://docs.google.com/dsl/org.gradle.api.publish.ivy.tasks.GenerateIvyDescriptor.html)

Creates an Ivy descriptor file for the publication named *PubName*, populating the known metadata such as project name, project version, and the dependencies. The default location for the descriptor file is *build/publications/$pubName/ivy.xml*.

publish*PubName*PublicationTo*RepoName*Repository — [PublishToIvyRepository](http://docs.google.com/dsl/org.gradle.api.publish.ivy.tasks.PublishToIvyRepository.html)

Publishes the *PubName* publication to the repository named *RepoName*. If you have a repository definition without an explicit name, *RepoName* will be "Ivy".

publish

*Depends on*: All publish*PubName*PublicationTo*RepoName*Repository tasks

An aggregate task that publishes all defined publications to all defined repositories.

[Publications](#17dp8vu)

This plugin provides [publications](http://docs.google.com/publishing_overview.html#glossary:publication) of type [IvyPublication](http://docs.google.com/dsl/org.gradle.api.publish.ivy.IvyPublication.html). To learn how to define and use publications, see the section on [basic publishing](http://docs.google.com/publishing_overview.html#sec:basic_publishing).

There are four main things you can configure in an Ivy publication:

* A [component](http://docs.google.com/publishing_overview.html#glossary:component) — via [IvyPublication.from(org.gradle.api.component.SoftwareComponent)](http://docs.google.com/dsl/org.gradle.api.publish.ivy.IvyPublication.html#org.gradle.api.publish.ivy.IvyPublication:from(org.gradle.api.component.SoftwareComponent)).
* [Custom artifacts](http://docs.google.com/publishing_overview.html#sec:publishing_custom_artifacts_to_maven) — via the [IvyPublication.artifact(java.lang.Object)](http://docs.google.com/dsl/org.gradle.api.publish.ivy.IvyPublication.html#org.gradle.api.publish.ivy.IvyPublication:artifact(java.lang.Object)) method. See [IvyArtifact](http://docs.google.com/dsl/org.gradle.api.publish.ivy.IvyArtifact.html) for the available configuration options for custom Ivy artifacts.
* Standard metadata like module, organisation and revision.
* Other contents of the module descriptor — via [IvyPublication.descriptor(org.gradle.api.Action)](http://docs.google.com/dsl/org.gradle.api.publish.ivy.IvyPublication.html#org.gradle.api.publish.ivy.IvyPublication:descriptor(org.gradle.api.Action)).

You can see all of these in action in the [complete publishing example](#26in1rg). The API documentation for IvyPublication has additional code samples.

[Identity values for the published project](#35nkun2)

The generated Ivy module descriptor file contains an <info> element that identifies the module. The default identity values are derived from the following:

* organisation - [Project.getGroup()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:group)
* module - [Project.getName()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:name)
* revision - [Project.getVersion()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:version)
* status - [Project.getStatus()](http://docs.google.com/dsl/org.gradle.api.Project.html#org.gradle.api.Project:status)
* branch - (not set)

Overriding the default identity values is easy: simply specify the organisation, module or revision properties when configuring the [IvyPublication](http://docs.google.com/dsl/org.gradle.api.publish.ivy.IvyPublication.html). status and branch can be set via the descriptor property — see [IvyModuleDescriptorSpec](http://docs.google.com/dsl/org.gradle.api.publish.ivy.IvyModuleDescriptorSpec.html).

The descriptor property can also be used to add additional custom elements as children of the <info> element, like so:

[Example: customizing the publication identity](#1ksv4uv)

**build.gradle**

publishing {  
 publications {  
 ivy(IvyPublication) {  
 organisation = 'org.gradle.sample'  
 module = 'project1-sample'  
 revision = '1.1'  
 descriptor.status = 'milestone'  
 descriptor.branch = 'testing'  
 descriptor.extraInfo 'http://my.namespace', 'myElement', 'Some value'  
  
 from components.java  
 }  
 }  
}

| **💡** | Certain repositories are not able to handle all supported characters. For example, the : character cannot be used as an identifier when publishing to a filesystem-backed repository on Windows. |
| --- | --- |

Gradle will handle any valid Unicode character for organisation, module and revision (as well as the artifact’s name, extension and classifier). The only values that are explicitly prohibited are \, / and any ISO control character. The supplied values are validated early during publication.

[Customizing the generated module descriptor](#44sinio)

At times, the module descriptor file generated from the project information will need to be tweaked before publishing. The Ivy Publish Plugin provides a DSL for that purpose. Please see [IvyModuleDescriptorSpec](http://docs.google.com/dsl/org.gradle.api.publish.ivy.IvyModuleDescriptorSpec.html) in the DSL Reference for the complete documentation of available properties and methods.

The following sample shows how to use the most common aspects of the DSL:

[Example: Customizing the module descriptor file](#2jxsxqh)

**build.gradle**

publications {  
 ivyCustom(IvyPublication) {  
 descriptor {  
 license {  
 name = 'The Apache License, Version 2.0'  
 url = 'http://www.apache.org/licenses/LICENSE-2.0.txt'  
 }  
 author {  
 name = 'Jane Doe'  
 url = 'http://example.com/users/jane'  
 }  
 description {  
 text = 'A concise description of my library'  
 homepage = 'http://www.example.com/library'  
 }  
 }  
 }  
}

In this example we are simply adding a 'description' element to the generated Ivy dependency descriptor, but this hook allows you to modify any aspect of the generated descriptor. For example, you could replace the version range for a dependency with the actual version used to produce the build.

You can also add arbitrary XML to the descriptor file via [IvyModuleDescriptorSpec.withXml(org.gradle.api.Action)](http://docs.google.com/dsl/org.gradle.api.publish.ivy.IvyModuleDescriptorSpec.html#org.gradle.api.publish.ivy.IvyModuleDescriptorSpec:withXml(org.gradle.api.Action)), but you can not use it to modify any part of the module identifier (organisation, module, revision).

| **⚠** | It is possible to modify the descriptor in such a way that it is no longer a valid Ivy module descriptor, so care must be taken when using this feature. |
| --- | --- |

[Repositories](#3rdcrjn)

This plugin provides [repositories](http://docs.google.com/dependency_management_terminology.html#sub:terminology_repository) of type [IvyArtifactRepository](http://docs.google.com/dsl/org.gradle.api.artifacts.repositories.IvyArtifactRepository.html). To learn how to define and use repositories for publishing, see the section on [basic publishing](http://docs.google.com/publishing_overview.html#sec:basic_publishing).

Here’s a simple example of defining a publishing repository:

[Example: Declaring repositories to publish to](#z337ya)

**build.gradle**

publishing {  
 repositories {  
 ivy {  
 // change to point to your repo, e.g. http://my.org/repo  
 url = "$buildDir/repo"  
 }  
 }  
}

The two main things you will want to configure are the repository’s:

* URL (required)
* Name (optional)

You can define multiple repositories as long as they have unique names within the build script. You may also declare one (and only one) repository without a name. That repository will take on an implicit name of "Ivy".

You can also configure any authentication details that are required to connect to the repository. See [IvyArtifactRepository](http://docs.google.com/dsl/org.gradle.api.artifacts.repositories.IvyArtifactRepository.html) for more details.

[Complete example](#26in1rg)

The following example demonstrates publishing with a multi-project build. Each project publishes a Java component and a configured additional source artifact. The descriptor file is customized to include the project description for each project.

[Example: Publishing a Java module](#3j2qqm3)

**build.gradle**

subprojects {  
 apply plugin: 'java'  
 apply plugin: 'ivy-publish'  
  
 version = '1.0'  
 group = 'org.gradle.sample'  
  
 repositories {  
 mavenCentral()  
 }  
 task sourcesJar(type: Jar) {  
 from sourceSets.main.java  
 classifier = 'sources'  
 }  
}  
  
project(':project1') {  
 description = 'The first project'  
  
 dependencies {  
 compile 'junit:junit:4.12', project(':project2')  
 }  
}  
  
project(':project2') {  
 description = 'The second project'  
  
 dependencies {  
 compile 'commons-collections:commons-collections:3.2.2'  
 }  
}  
  
subprojects {  
 publishing {  
 repositories {  
 ivy {  
 // change to point to your repo, e.g. http://my.org/repo  
 url = "${rootProject.buildDir}/repo"  
 }  
 }  
 publications {  
 ivy(IvyPublication) {  
 from components.java  
 artifact(sourcesJar) {  
 type = 'sources'  
 conf = 'compile'  
 }  
 descriptor.description {  
 text = description  
 }  
 }  
 }  
 }  
}

The result is that the following artifacts will be published for each project:

* The Ivy module descriptor file: ivy-1.0.xml.
* The primary JAR artifact for the Java component: project1-1.0.jar.
* The source JAR artifact that has been explicitly configured: project1-1.0-source.jar.

When project1 is published, the module descriptor (i.e. the ivy.xml file) that is produced will look like:

[Example: Generated ivy.xml](#1y810tw)

**output-ivy.xml**

<!-- This file is an example of the Ivy module descriptor that this build will produce -->  
<?xml version="1.0" encoding="UTF-8"?>  
<ivy-module version="2.0" xmlns:m="http://ant.apache.org/ivy/maven">  
 <info organisation="org.gradle.sample" module="project1" revision="1.0" status="integration" publication="«PUBLICATION-TIME-STAMP»">  
 <description>The first project</description>  
 </info>  
 <configurations>  
 <conf name="compile" visibility="public"/>  
 <conf name="default" visibility="public" extends="compile,runtime"/>  
 <conf name="runtime" visibility="public"/>  
 </configurations>  
 <publications>  
 <artifact name="project1" type="sources" ext="jar" conf="compile" m:classifier="sources"/>  
 <artifact name="project1" type="jar" ext="jar" conf="compile"/>  
 </publications>  
 <dependencies>  
 <dependency org="junit" name="junit" rev="4.12" conf="compile->default"/>  
 <dependency org="org.gradle.sample" name="project2" rev="1.0" conf="compile->default"/>  
 </dependencies>  
</ivy-module>

| **💡** | Note that «PUBLICATION-TIME-STAMP» in this example Ivy module descriptor will be the timestamp of when the descriptor was generated. |
| --- | --- |

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/)